AFRL-AFOSR-UK-TR-2011-0038

Effective Task Assignment and Motion Planning for Complex UAV Operations

Tal Y. Shima

Technion - Israel Institute of Technology
Faculty of Aerospace Engineering
Technion City
Haifa, Israel 32000

EOARD GRANT 09-3066

September 2011

Final Report for 31 August 2009 to 31 August 2011

Distribution Statement A: Approved for public release distribution is unlimited.

Air Force Research Laboratory
Air Force Office of Scientific Research
European Office of Aerospace Research and Development
Unit 4515 Box 14, APO AE 09421

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. REPORT DATE ***(DD-MM-YYYY)***	**2. REPORT TYPE**	**3. DATES COVERED** *(From – To)*
20-09-2011	Final Report	31 August 2009 – 31 August 2011

4. TITLE AND SUBTITLE

Effective Task Assignment and Motion Planning for Complex UAV Operations

5a. CONTRACT NUMBER

FA8655-09-1-3066

5b. GRANT NUMBER

Grant 09-3066

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Professor Tal Y. Shima

5d. PROJECT NUMBER

5d. TASK NUMBER

5e. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Technion - Israel Institute of Technology
Faculty of Aerospace Engineering
Technion City
Haifa, Israel 32000

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

EOARD
Unit 4515 BOX 14
APO AE 09421

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL/AFOSR/RSW (EOARD)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

AFRL-AFOSR-UK-TR-2011-0038

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This two years research effort was concerned with the efficient solution of combinatorial optimization problems that arise in the task assignment and motion planning of unmanned aerial vehicles in complex operations. In the first year of the research we have concentrated on the high level task assignment problem. A genetic algorithm that utilizes process algebra for coding of solution chromosomes and for defining evolutionary based operators was presented. The algorithm is applicable to a general class of mission planning and optimization problems. In the second year of the research we paid special attention to the interaction between the high level mission planning (task assignment) and the low level motion-planning for unmanned aerial vehicles. We considered an unmanned aerial vehicle modeled as a Dubins vehicle: a vehicle with a minimum turn radius and the inability to go backward. Given a starting position and orientation for a Dubins vehicle and a set of stationary targets, the main problem is to determine the shortest flyable path that visits each target. This problem is called the Dubins traveling salesman problem, an extension of the well-known traveling salesman problem. A number of algorithms with different approaches, including a hierarchical approach, a generalized traveling salesman problem reformulation approach, and a search with an upper bound Dubins cost approach, is developed and contrasted. Monte Carlo simulations were performed for a range of vehicle turn radii. Simulations results show that integrating two plausible kinematic satisfying paths as an upper bound to determine the cost-so-far into a search algorithm generally improves performance in terms of the shortest path cost and search complexity. Furthermore, when a suitable trajectory for traversing an ordered set of targets has been found, the applicability of using proportional navigation guidance is examined. It is shown that using a simple proportional navigation guidance law, stationary targets located within the vehicle's turn circle are unreachable.

15. SUBJECT TERMS

EOARD, Flight Control, Unmanned Vehicles, Numerical Simulation

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18, NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Gregg Abate
UNCLAS	UNCLAS	UNCLAS	SAR	38	**19b. TELEPHONE NUMBER** *(Include area code)* +44 (0)1895 616021

Faculty of Aerospace Engineering
Technion - Israel Institute of Technology

Effective Task Assignment and Motion Planning for Complex UAV Operations

AFOSR Contract No. FA8655-09-1-3066

Final Report
(Covering the period: 31 August 2009 - 30 August 2011)

Principal Investigator: Dr. Tal Shima

Abstract

This two years research effort was concerned with the efficient solution of combinatorial optimization problems that arise in the task assignment and motion planning of unmanned aerial vehicles in complex operations.

In the first year of the research we have concentrated on the high level task assignment problem. A genetic algorithm that utilizes process algebra for coding of solution chromosomes and for defining evolutionary based operators was presented. The algorithm is applicable to a general class of mission planning and optimization problems. As an example the high level mission planning for a cooperative group of uninhabited aerial vehicles was investigated. The mission planning problem was cast as an assignment problem, and solutions to the assignment problem were given in the form of chromosomes that are manipulated by evolutionary based operators. The evolutionary operators of crossover and mutation were formally defined using the process algebra methodology, along with specific algorithms needed for their execution. The viability of the approach was investigated using simulations and the effectiveness of the algorithm was shown in small, medium, and large scale problems. The work, performed in collaboration with Prof. Emilio Frazzoli and Sertac Karaman of MIT, was presented in last year's annual report.

In the second year of the research we paid special attention to the interaction between the high level mission planning (task assignment) and the low level motion-planning for unmanned aerial vehicles. This effort is described in the current report. We considered an unmanned aerial vehicle modeled as a Dubins vehicle: a vehicle with a minimum turn radius and the inability to go backward. Given a starting position and orientation for a Dubins vehicle and a set of stationary targets, the main problem is to determine the shortest flyable path that visits each target. This problem is called the Dubins traveling salesman problem, an extension of the well-known traveling salesman problem. A number of algorithms with different approaches, including a hierarchical approach, a generalized traveling salesman problem reformulation approach, and a search with an upper bound Dubins cost approach, is developed and contrasted. Monte Carlo simulations were performed for a range of vehicle turn radii. Simulations results show that integrating two plausible kinematic satisfying paths as an upper bound to determine the cost-so-far into a search algorithm generally improves performance in terms of the shortest path cost and search complexity. Furthermore, when a suitable trajectory for traversing an ordered set of targets has been found, the applicability of using proportional navigation guidance is examined. It is shown that using a simple proportional navigation guidance law, stationary targets located within the vehicle's turn circle are unreachable. To address this issue, waypoints are generated that will allow the vehicle to reach these hard targets with minimum distance. The waypoints are generated by solving a relaxed version of the point to point Dubins problem. In addition, it is shown that any trajectory composed of point to point Dubins paths can be navigated using proportional navigation with a minimum number of waypoints, regardless of the target positions. The research presented in this report was performed as part of the M.Sc. work of my graduate student Matt Cons.

The results of the first year of the research have been accepted for journal publication, see:

- Sertac, K., Shima, T., and Frazzoli, E., ”A Process Algebra Genetic Algorithm", IEEE Transactions on Evolutionary Computation.

The results of the second year of the research have recently been presented in a conference and published in its proceedings, see:

- Cons, M., Shima, T., and Domshlak, C., ”Integrating Task Assignment and Guidance via Motion Planning for Unmanned Aerial Vehicles", Proceedings of the AIAA Guidance, Navigation, and Control Conference, Portland, OR, August, 2011.

Contents

List of Figures

I. Introduction

Interest in unmanned aerial vehicles (UAVs) has grown considerably as they can be used for a number of applications, including remote sensing[1], surveillance [2], search and rescue operations [3], and cooperative control of multiple UAVs [4]. A common problem to many of these applications is how the UAV traverses over multiple targets in the shortest distance or time, a problem strongly resembling the traveling salesman problem (TSP). Because many UAVs are fixed-wing aircraft, they have kinematic constraints that make this problem more difficult. When the vehicle is modeled as a Dubins vehicle, the problem is called the Dubins TSP (DTSP) [5]. Once a trajectory has been proposed, an appropriate guidance law must be implemented. This is more difficult with kinematic constraints.

The DTSP was so named because of the results found by Dubins who analyzed the shortest path problem for a nonholonomic vehicle in 1957. Specifically, Dubins analyzed the shortest path for what is now called a Dubins vehicle: a vehicle that has the kinematic constraints of a minimum turn radius and the inability to go backward [6, 7, 5]. Dubins provided the solution to the shortest path problem between two states $(x_i, y_i, \theta_i) \rightarrow (x_f, y_f, \theta_f)$, including the coordinate position and vehicle heading (or orientation) [6]. The shortest path must be one of the following six different combinations of line segments and curvature arcs: RLR, LRL, RSR, LSL, RSL, LSR, where R is a right hand turn (clockwise), L is a left hand turn (counterclockwise), and S is a straight line [6]. The turns are along the vehicle's minimum turn radius.

Whereas Dubins solved the shortest path problem from one point with given orientation to a second point with given orientation, Ma and Castañón [8] consider and solve a variation to the Dubins shortest path problem for the case where the second point is given but not the orientation $(x_i, y_i, \theta_i) \rightarrow (x_f, y_f)$ - that is, the orientation of the second point is free. They defined the problem as a minimum time trajectory problem, defined the Hamiltonian, and used the Pontryagin's Maximum Principle to solve for the optimal path. It was determined that the optimal trajectories consist of four types: RS, LS, RL, and LR segments. For the cases of RL and LR types, the last arc curvature is greater than π radians. There are fewer types than described by Dubins because the terminal orientation constraint has been dropped [8]. Enright et al. later gave the analytical solution [9].

Some research has been conducted to investigate the DTSP because of its applications toward UAVs. Unlike the Euclidean TSP (ETSP), in which the distances between targets is calculated from the Euclidean metric, it is not possible to use classical combinatorial optimization methods on the DTSP because the problem cannot be defined as a finite graph [10]. At each target, the state of the vehicle includes the vehicle's heading angle, which is not included in an ETSP, for example. As the heading angle can be described anywhere between $[0, 2\pi]$, the number of heading angle possibilities is infinite.

Some researchers have chosen to address the DTSP in a hierarchical approach as shown in Figure 1 [10, 8]. They assume that the ordering of the targets has already been given or the corresponding ETSP has been solved such that the Dubins constraints have been dropped. Assigning the order of targets is a form of task planning. Given the ordering, the second step is to propose a trajectory that satisfies the kinematic constraints of the vehicle, a type of motion planning. For example, Salva et al. propose an algorithm, the Alternating Algorithm, for a Dubins tour, a tour through an ordered set of vertexes satisfying the Dubins vehicle constraints [10]. The path produced by the Alternating Algorithm consists of alternating straight line segments and Dubins path segments [10].

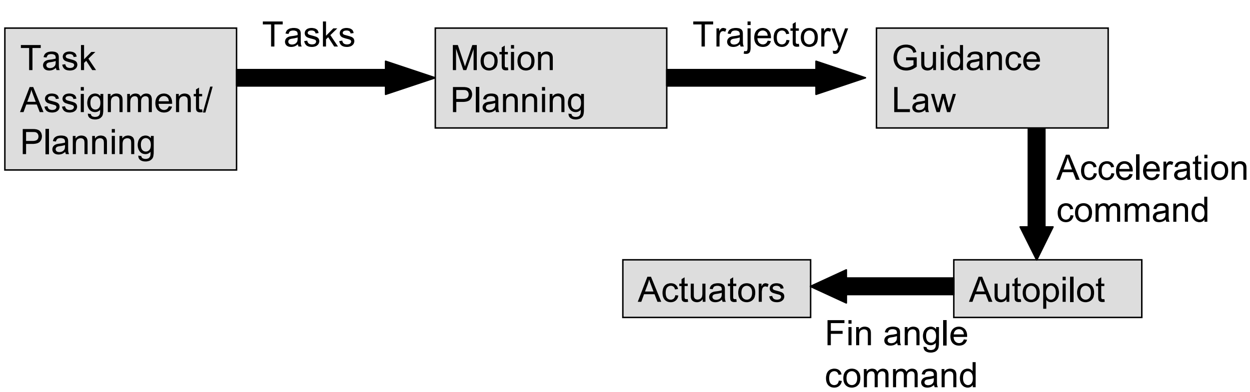


Figure 1. Possible hierarchical problem approach

Another algorithm for path planning given an ordered set of vertexes is called the Two-point Algorithm [8]. Given the initial vehicle position and heading angle, the trajectory to the first target is calculated using the relaxed Dubins solution. The relaxed Dubins solution gives the angle over the first target. Now the trajectory to the second target can be calculated using the first target and its newly assigned heading angle. The process is continued in a receding horizon manner until all targets have been reached. For clarity, this work refers to this algorithm as the relaxed Dubins algorithm.

A more integrated approach for task planning and motion planning was given by Edison and Shima [11]. They approximated the DTSP as a generalized traveling salesman problem (GTSP), which is an extension of the TSP. The GTSP can be stated as follows: Given a starting position and a number of clusters, each containing a number of target locations, find the shortest path which passes through exactly one target location from each cluster and returns to the start position [12]. Edison and Shima reformulated the problem by uniformly discretizing the possible vehicle heading angles over each target, which constitutes the clusters in the GTSP. The interest was to assign tasks and plan paths for a group of cooperative UAVs. The problem was addressed by constructing a graph and by implementing a deterministic search and a stochastic search via genetic algorithms.

TSP is a classic example of a problem that belongs to the NP-complete complexity group [13]. Garey et al. and Papadimitriou showed that the ETSP with finite precision is NP-complete by reducing the ETSP to the Exact Cover problem, an NP-complete problem [14, 15]. However, they noted that the question of whether or not the ETSP falls into NP, and thus NP-complete, remains open because of precision considerations of using the Euclidean metric, e.g. calculating square roots [14, 15]. Papadimitriou defined the Euclidean-tour TSP and the Euclidean-path TSP and found them to be NP-complete, ignoring the precision consideration [15]. The Euclidean-tour TSP demands a tour, a path that returns to the starting point, whereas the Euclidean-path TSP does not [15]. Ny et al. considered the DTSP and showed that the tour-DTSP and the path-DTSP are both NP-hard by extending the proof of Papadimitriou[16]. The focus was on the hardness of the DTSP because of similar concerns regarding precision in calculating Dubins tours, e.g. calculating trigonometric functions [16].

The main contribution of this work is that it proposes and compares different algorithms that integrate the task planning and the motion planning aspects of the problem, rather than treating the two separately. This work proposes using an upper bound on calculating kinematic satisfying paths for setting costs in the search algorithm. This work compares the proposed algorithm to algorithms that solve the search aspect first and then solve the motion planning aspect second. In addition, once an ordered set of stationary targets has been selected, possibly through the method shown in this work, the use of proportional navigation guidance for traversing an ordered set of stationary targets in minimum distance is analyzed and implemented with small but important modifications.

Section II briefly describes the problem formulation, including the vehicle constraints. Section III describes the primary search method used. This section also expounds on the different algorithms that combine the task planning and the vehicle dynamics. Section IV illustrates and discusses the results of these algorithms. Section V describes how one could use the proportional navigation guidance law to traverse through the targets, once an ordering of visitation has been chosen. Section VI briefly summarizes the main results.

II. Problem Formulation

This section summarizes the main problem of this work through a mathematical formulation. The vehicle constraints are defined as well.

II.A. Search

This research looks to address the DTSP, which has the requirements of a classical TSP, but also includes kinematic constraints on the "salesman," or vehicle, as described in Sub-Section II.C. Similar to a TSP, given a starting position, the vehicle must visit each of the V targets, or vertexes, once and only once in a closed tour. However, in this research, the vehicle does not have to return to the starting position. This is called the path-DTSP [16].

II.B. Optimization problem

The optimization problem can be stated concisely as minimizing the cost function J, the cost of the path,

$$\min J = \sum_{i=1}^{N} \sum_{j=2}^{N} X_{(m_i,m_j)} d_{(m_i,m_j)}$$

where $i \neq j$, $m_i \in M$, in which M is the set of all target locations plus the initial vehicle location, $X_{(m_i,m_j)} \in \{0,1\}$ are decision variables, $d_{(m_i,m_j)}$ is the Dubins distance from m_i to m_j, and N=|M|. In this formulation, associated with each m in M is a free vehicle heading angle from $[0, 2\pi]$, which must be selected. When the heading angle for each m is defined, then the Dubins distance d can be calculated.

This minimization is under the constraints of a Dubins vehicle shown in Sub-Section II.C. In addition, as shown by Edison and Shima, the set D

$$D = \{e_{m_i,m_j} | (X_{m_i,m_j} = 1), m_i \in M, m_j \in M\}$$

must compose a complete path of edges e to the targets starting from the initial vehicle condition [11].

II.C. Vehicle constraints

To be applicable to UAVs, the vehicle's dynamics are constrained to that of a Dubins vehicle that has a constant positive speed [7]. The equations of motion for a Dubins vehicle are presented below:

$$\begin{aligned} \dot{x} &= s\cos\theta \\ \dot{y} &= s\sin\theta \\ \dot{\theta} &= \Omega_{\max} u \\ \dot{s} &= 0 \end{aligned}$$

where x and y are Cartesian coordinates, s is the speed, θ is the azimuth flight heading, and $\Omega_{\max}$ is the maximum turn rate. The controller u is an element of the set $U = \{-1, 0, 1\}$, corresponding to turn right, go straight, turn left, respectively.

III. Search Algorithms

Several search algorithms for finding the shortest path for the DTSP are detailed next. They can be divided into 3 categories: ETSP with overlaying kinematic constraint satisfying solution, GTSP reformulation, and TSP with Dubins cost function. The algorithms all employ a depth first search method.

III.A. ETSP with overlaying kinematic constraint satisfying solution (hierarchical approach)

The first class of algorithms represents the hierarchical approach as illustrated in the first two blocks of Figure 1. The first step is to decide the order of visitation and then determine a trajectory that satisfies the kinematic constraints of the vehicle. In the first step, the kinematic constraints are dropped, and the corresponding ETSP is solved, providing the ordering of the vertexes. Given the ordering, the second step is to provide a trajectory that satisfies the kinematic constraints of the Dubins vehicle. This research investigates two methods for providing a flyable trajectory: the relaxed Dubins algorithm proposed by Ma and Castañón [8] and the Alternating Algorithm proposed by Savla et al. [10] This set of algorithms can be considered a benchmark to compare the other two classes of algorithms, since these methods represent the hierarchical approach.

III.A.1. Relaxed Dubins algorithm method

Given the order of visitation from the corresponding ETSP solution, the relaxed Dubins method is defined as follows. Given the initial vehicle state, position and heading, the distance and heading angle over the first target is determined by solving the relaxed Dubins problem. Recall that the relaxed Dubins problem is the problem of finding the minimum distance between a point and orientation to another point without regard to orientation, i.e. $(x_i, y_i, \theta_i) \to (x_f, y_f)$. Now that the heading angle over the first target has been defined, the relaxed Dubins method can be solved again to the second target. This continues until all the vertexes have been assigned angles. The total distance of the path can be calculated easily using the results of the

relaxed Dubins solution or summing the result of the point to point Dubins solution for each segment in the path.

III.A.2. Alternating Algorithm method

Given the order of visitation from the corresponding ETSP solution, the Alternating Algorithm is utilized to determine the angles of each vertex. Once the angles of each vertex are set, the total distance of the path can be calculated easily by summing the point to point Dubins solution for each segment in the path. As mentioned earlier, the Alternating Algorithm consists of alternating straight line segments and Dubins path segments. The Alternating Algorithm used in this research is slightly modified in that the first segment from the initial start position to the first target is always a curved Dubins path segment, and the following segment is a straight line path between the first and second target. This slight change is due to the randomized initial vehicle state, including the initial orientation of the vehicle, which is independent of the target locations. It is not normally possible to set a straight line path from the initial vehicle state to the first target.

III.B. Generalized traveling salesman problem (GTSP)

The DTSP can be approximated and reformulated into a GTSP by discretizing the vehicle's possible heading angle over each target (vertex). See Figure 2. This is a relaxed problem because the possible angle states that the vehicle can be in are discretized, compared to the infinite angle possibilities between $[0, 2\pi]$ in the DTSP. Once this step is complete, a search to find the optimal path for the GTSP is performed. The distances between states (position and heading angle) are determined using the point to point Dubins solution. Unlike the hierarchical approach, the search tree for this approach must include discretized angle possibilities as well as the targets (Figure 3).

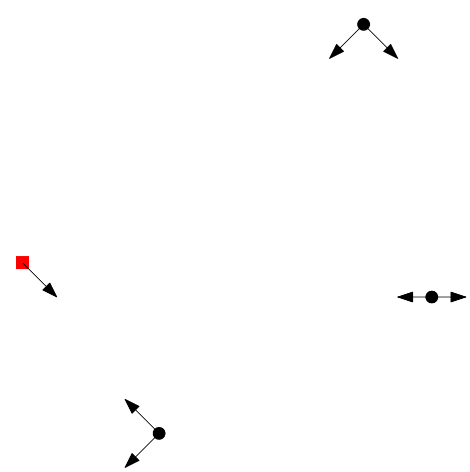


Figure 2. An approximation of the DTSP into a GTSP reformulation. There are 3 cities to be visited. Here, the number of cities and the values of the angles were chosen arbitrarily for illustration purposes. The initial position of the vehicle is represented as a square, and its initial heading angle is shown. A search algorithm must choose the ordering of the vertexes and one heading angle for each target.

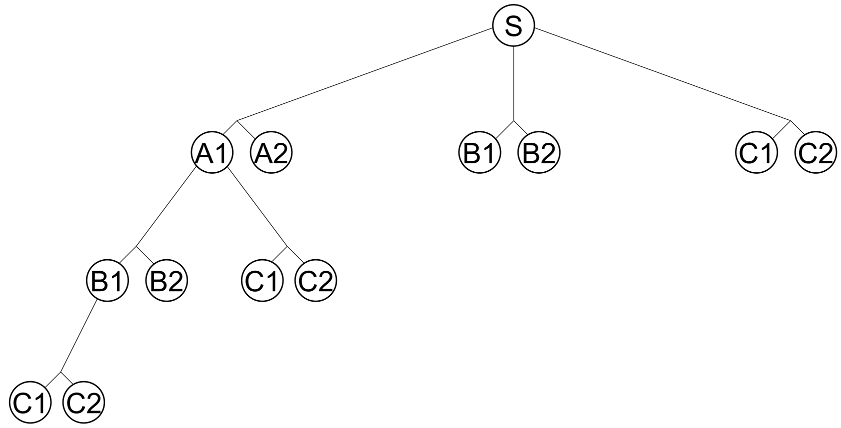


Figure 3. Search tree of an approximation of the DTSP into a GTSP reformulation, corresponding to Figure 2. A, B, and C represent targets. The numbers represent angle possibilities for the corresponding targets. The GTSP tree is much larger than its TSP counterpart and only the left most part of the tree is shown, down to the leaf node.

While the number of angle candidates for every vertex does not have to be the same, in this research the number of angles is indeed the same for every vertex. It may be better to choose the number of angles over each target in a non uniform manner. The values of the angle candidates were chosen in a few manners described here.

III.B.1. GTSP with uniform angle candidates

In this algorithm, the DTSP was reformulated into a GTSP by discretizing the possible angles over each vertex. The number of angles is the same for each vertex. The values of the angle candidates were chosen in a uniform manner. This algorithm was proposed and investigated by Edison and Shima [11]. For example, if the number of angle candidates is one, then every angle candidate has the value of $\{0\}$. If the number of angle candidates is four, then the angle candidates have the values of $\left\{0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}\right\}$ (Figure 4).

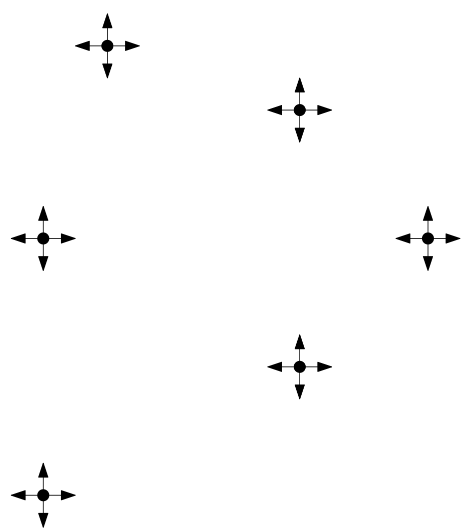


Figure 4. A simple reformulation of a DTSP into a GTSP. There are 6 targets to be visited. Here, the number of angle candidates of the vertexes was set at 4, while their values were chosen uniformly. Their values consists of $\left\{0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}\right\}$.

III.B.2. GTSP with random angle candidates

In this algorithm, the DTSP was reformulated into a GTSP by discretizing the possible angles over each vertex. The number of angles is the same for each vertex. The values of the angle candidates were chosen

randomly from $[0, 2\pi]$.

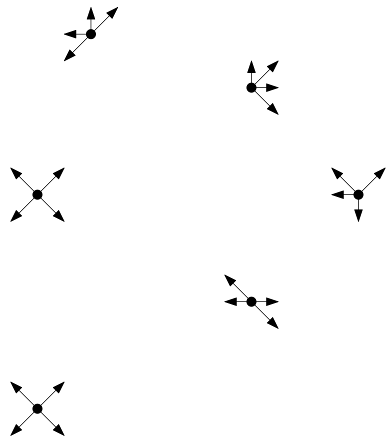


Figure 5. A simple reformulation of a DTSP into a GTSP. There are 6 targets to be visited. Here, the number of angle candidates of the vertexes was set to 4, while their values were chosen randomly.

III.B.3. GTSP with straight line angles to nearest neighbors

This algorithm was inspired by the idea that it may be advantageous to travel in a straight line to the nearest target. Given N candidate angles per vertex, for each m vertex, the $\frac{\mathrm{N}}{2}$ closest neighbors c are considered. For each m, the values of the N candidate angles are assigned according to the straight line angle from m to each c and from each c to m. The algorithm calls for the $\frac{\mathrm{N}}{2}$ closest neighbors because it is desired to give the opportunity to travel in a straight line from m to c or c to m, not just in one direction.

Figure 6. For GTSP with straight line angles to nearest neighbors. There are 6 targets to be visited. The number of candidate angles $\mathrm{N} = 4$ in this example. The angles for each vertex were set according to the straight line angles to the 2 closest neighbors.

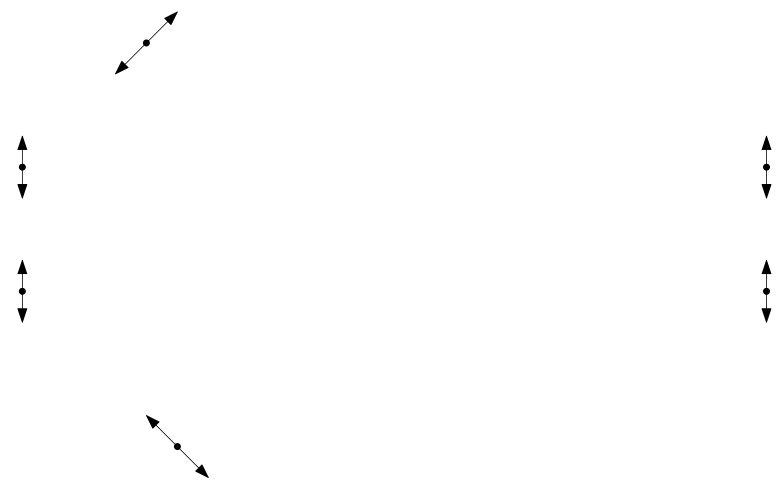


Figure 7. For GTSP with straight line angles to nearest neighbors. There are 6 targets to be visited. The number of candidate angles N = 2. The angles for each vertex were set according to the straight line angle to the closest neighbor.

III.C. TSP with Dubins cost function

This research proposes using kinematic-constraint-satisfying paths to define an upper bound on the cost of partial paths, the cost-so-far $g(n)$, within the search algorithm. Because the costs are defined through motion planning, the search integrates both task planning and motion planning. In the following algorithms, the search will take the form of a typical TSP search; however, the method of calculating $g(n)$ will differ for each depending on the upper bound chosen. Because there is no known way to calculate the minimum $g(n)$ for a Dubins path given multiple targets, the proposed $g(n)$ functions do not provide a minimum cost, an unusual characteristic of a TSP. The heuristic used for all algorithms is the Euclidean minimum spanning tree.

III.C.1. TSP relaxed Dubins search

When a new vertex is added to a partial path, its cost and its final heading angle are determined by the relaxed Dubins solution. This algorithm is very similar to the ETSP with overlaying relaxed Dubins algorithm of the hierarchical approach. Here, however, the costs of satisfying the kinematic constraints is integrated within the search.

III.C.2. TSP Alternating Algorithm (AA) search

This method implements the AA concept. The vehicle cost from the start vertex to the second vertex is determined by the Euclidean distance between them. Due to the fact that the third vertex in the path is currently unknown, the second vertex angle cannot be calculated yet. When a new vertex is added to the path, such that there are three total vertexes, the heading angle of the second vertex is the straight line angle between the second vertex and the third vertex. The third vertex heading angle is equal to the second vertex. The cost between the first and second vertex is now updated and is calculated by the Dubins path for $(x_{start}, y_{start}, \theta_{start}) \rightarrow (x_{second}, y_{second}, \theta_{second})$. The cost between the second vertex to the third vertex is the Euclidean distance. Similar to the situation of adding the second vertex to the path, when the fourth vertex is added to the path, the heading angle for this vertex cannot be set until the fifth vertex is added. Thus, the partial path of the fourth vertex is the cost of the path of three vertexes plus the Euclidean distance between the third and fourth vertexes.

Using this algorithm, the cost of partial paths with an even number of vertexes may be underestimated compared to the optimal Dubins cost of the same partial path given the angles from the start vertex through the second to last vertex. This will not cause a problem in the search. Only partial paths that are assigned costs that exceed their true cost risk being pruned from the search. This is similar to the concept of an admissible heuristic.

III.C.3. TSP joint AA and relaxed Dubins search

This method aims to combine the two above methods: AA and relaxed Dubins. It works as follows: The vehicle travels from the start vertex to the second vertex added according to the relaxed Dubins solution. When the number of vertexes in the partial path is even, the last vertex is assigned the angle solved from the relaxed Dubins problem between the second-to-last vertex and the last vertex. Thus, when the partial path has exactly 3 vertexes, this algorithm is the same as Relaxed Dubins TSP search. When the number of vertexes in the partial path is odd, then two possibilities are examined and their costs determined. The smaller cost of the two methods will be used. The first method is using the relaxed Dubins between the second to last vertex and the last vertex. The second method is to determine the straight line angle between the second to last vertex and the last vertex and assign both vertexes this angle. This entails changing the second to last vertex's angle - this is rather unusual as it effectively changes part of the partial path that does not include the most recent vertex added via the search.

It is expected that this algorithm will outperform the other TSP with Dubins cost function search algorithms because when a new vertex is added to the partial path, two methods to determine the new angle are investigated and the best one is chosen. However, this algorithm does not necessarily dominate the other two; that is, this algorithm does not guarantee to find a better solution as it works on a greedy method.

III.D. Summary of algorithms

To summarize, there are eight total search algorithms, which fall into three major classes or approaches. The ETSP/AA and the ETSP/Relaxed Dubins fall into the hierarchical approach. The GTSP algorithms, including GTSP uniform, GTSP random, and GTSP straight line angles, are reformulations of an approximation to the DTSP. The final class, including the TSP Relaxed Dubins search, TSP AA Search, and the TSP Joint AA and Relaxed Dubins Search, use an upper bound to calculate the Dubins cost along the search tree. The last two approaches are integrated approaches as they take into account the motion planning while performing a search; however, the last approach maintains the complexity of the search to a TSP rather than a GTSP. The algorithms are summarized in Table 1. A comparison of how $g(n)$ is calculated in the search tree is shown in Figure 8. In the hierarchical approach, the cost of a path is calculated by summing the point to point Euclidean distances. In the GTSP approach, the cost is calculated by summing the point to point Dubins distances. In the TSP with Dubins upper bound, the cost is calculated based on a particular Dubins upper bound method for the entire length of the path.

Furthermore, while none of the algorithms guarantee a global optimal solution (in part because the analytical solution for the optimal path that traverses over multiple targets is unknown), all the algorithms are point to point optimal. They each use the point to point Dubins solution or point to point relaxed Dubins solution that provides the shortest path between two consecutive targets along the trajectory.

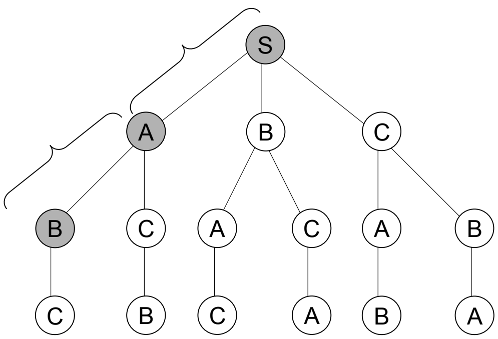
Table 1. Summary of search algorithms

Hierarchical Approach	ETSP/RelaxedD	Ma and Castañón (2006)
	ETSP/AA	Savla, Frazzoli, and Bullo (2005)
Integrated Approach - GTSP reformulation	GTSP uniform	Edison and Shima (2010)
	GTSP random	
	GTSP straight line angles	
Integrated Approach - Dubins costs	TSP AA Search	
	TSP Relaxed D Search	
	TSP Joint Relaxed D AA Search	

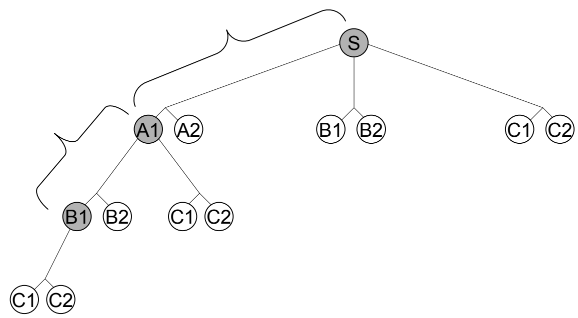
III.E. Search method - depth first search

The search method is based on a depth first search with pruning and an admissible heuristic.

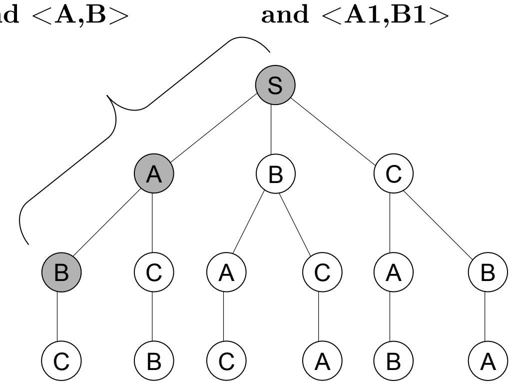
1. Initialization - The search is initialized by adding a given initial start node S to a queue of partial paths Q that are waiting to be investigated. Thus, the first partial path consists of the initial start node, or vertex, which includes the vehicle position and heading angle. Steps 2-6 run in a loop.



(a) Hierarchical approach - The cost of path <S,A,B> is calculated by summing the point to point Euclidean distances of <S,A> and <A,B>



(b) GTSP approach - The cost of path <S,A1,B1> is calculated by summing the point to point Dubins distances of <S,A1> and <A1,B1>



(c) TSP with Dubins upper bound approach - The cost of path <S,A,B> is calculated based on a Dubins upper bound for <S,A,B>

Figure 8. Comparison of the three classes of algorithms. Path in question is <S, A, B>.

2. Check time constraint - For some experiments, a time constraint was imposed on the algorithms. If the current elapsed time exceeds the given time constraint, the algorithm stops here and returns the best solution found so far. If not, or if no time constraint is given (the algorithm is allowed to run to completion), then move to the next step.

3. Remove from queue and prune if necessary - The next step is to remove the partial path P that is at the front of Q. The cost of the current partial path, C, is compared to the best solution found so far. If the cost of P is smaller, or there is no current best solution found so far, then P is moved to the next step. Otherwise, P is discarded and said to be pruned, and this step is repeated (the next partial path at the front of the queue is examined). If there are no more partial paths in Q, then the algorithm returns the best solution found so far.

4. Check solution - P is examined to determine whether or not it is a solution candidate (a complete path). If P is a solution candidate and its cost is less than current best solution (if one exists), then P now becomes the current best solution, and the previous best solution is discarded. If P is not a solution candidate, then it is moved to the next step.

5. Expanding paths - For each vertex v that is not in P, a new partial path is created which is v added to the end of P. The cost of each new partial path is calculated, and the paths are put in ascending order according to their costs. The details of this step depend on the specific algorithm used.

   (a) ETSP with Overlaying Kinematic Constraint Satisfying Solution - The $g(n)$ cost of the new partial path is P plus the Euclidean distance between the last vertex of P and v. The admissible heuristic $h(n)$, or cost-to-go, is the cost of the Euclidean MST for the unvisited vertexes plus v.

   (b) GTSP - In a GTSP search, not only is each vertex v that is not in P examined, but so are each of the candidate angles of v. Thus, for each vertex v and for every candidate angle α for v, a new partial path is created which is v with a added to the end of P. The g cost of the new partial path is P plus the Dubins distance between the last vertex of P and v with a. The admissible heuristic $h(n)$, or cost-to-go, is the cost of the Euclidean MST for the unvisited vertexes plus v.

   (c) TSP with Dubins cost function - Generally, the g cost of the new partial path is P plus the Dubins distance between the last vertex of P and v with an angle specified by the specific algorithm. If the angles of any of the vertexes in P are changed, then the new partial path's cost will need to reflect that change accordingly. The admissible heuristic $h(n)$, or cost-to-go, is the cost of the Euclidean MST for the unvisited vertexes plus v.

6. Adding expanded paths to the queue - The ordered expanded paths must be placed in the queue. In this research, the search algorithm is run in a depth first manner. This demands that the ordered expanded paths be put at the front of the queue.

IV. Monte Carlo Simulation Results

This section details the results of the Monte Carlo simulations for comparing the different algorithms. The setup for the simulations is explained first, followed by an explanation of how the algorithms are compared, and finally the results and analysis of the algorithms.

IV.A. Monte Carlo simulation setup

The vehicle and target positions were randomly generated in a 20 by 20 square centered at the origin. The vehicle's initial heading angle was randomly generated between $[0, 2\pi]$. Two datasets were created, one consisting of 6 targets and one consisting of 32 targets. Six is the largest number of targets possible such that all the search algorithms can run till completion in a reasonable amount of time using a specific simulation computer. Thirty-two is a large enough number of targets such that none of the algorithms solve to completion in 10 seconds. Thus, two situations were created, one in which the algorithms ran until completion and a second where none of the algorithms completed.

Each dataset consists of 100 trials. Through Monte Carlo simulations, it was found that the results did not significantly change after 20 trial runs. However, 100 trial runs were created for each dataset for added robustness.

It was hypothesized that the results may vary greatly depending on the vehicle turn radius. As the turn radius of the vehicle approaches zero, the solution should approach the ETSP solution. Thus, the algorithms were tested over the two datasets with radii varying from 0.01 to 1000. Because the vehicle's turn radius and the position of the targets and initial vehicle position are unitless, the results are scalable.

IV.B. Analysis method

The results of the algorithms are compared in three manners:

1. Quality Index: The quality index describes the percent improvement of one algorithm over the other algorithms. For a single trial, a score is calculated by taking the best result, which is the lowest cost, of the trial and assigning it as the base cost. The other algorithms receive a cost defined as $quality = \frac{cost_{base}}{cost_{algoirthm}}$. Then, the score for each algorithm of the 100 trials is summed, and this sum is the quality index score. Thus, an algorithm receiving a quality index score of 100 signifies that an algorithm produced the best path in each of the 100 trials. A score of 50 signifies that on average the algorithm produces a path that is twice as long as the best solutions from different algorithms.

2. Number of Winners: For each of the 100 simulations for a particular radius, 1 point is assigned to the algorithm that has the shortest path cost and zero otherwise. In the event of a tie of two or more algorithms, each algorithm is assigned 1 point. The points are summed to give a score. The score may be more than 100 total points when summed across all the algorithms because of possible ties. More points signify a better algorithm.

3. Average Cost: The average cost is the average path length that an algorithm produces over the 100 trials. Better algorithms will produce paths that have lower average costs.

Each approach can be illustrated as a function of turn radius or expanded paths, the number of partial paths that were removed from the search queue and expanded.

IV.C. Simulation results

The results are divided into two sections corresponding to the two datasets: a dataset of 100 trial setups with 6 targets and a dataset of 100 trial setups with 32 targets. In the following figures, the names of the algorithms are written with a condensed name. ETSP/RelaxedD and ETSP/AA are the hierarchical algorithms consisting of solving the corresponding ETSP and overlaying the relaxed Dubins algorithm and the Alternating Algorithm, respectively. GTSP uniform(4), GTSP random (4), GTSP straight line angles (4), represent the 3 different GTSP algorithms with 4 angles. TSP AA Search, TSP RelaxedD Search, and TSPJointRelaxedDAA Search, represent the three different TSP with Dubins cost function algorithms.

IV.C.1. Analysis of algorithms running to completion

The following describes the results of running the eight algorithms to completion for a dataset of 100 trials.

Figure 9 compares the eight algorithms as a function of the vehicle turn radius. By examining the left side of the graph for small turn radii, it is shown that the percent difference is negligible. Thus, all the algorithms provide a path with approximately the same cost. This result is expected since the DTSP converges to the ETSP as the turn radius of the Dubins vehicle approaches zero. A typical solution path for a small radius is shown in Figure 10a. Clearly, the path takes the form of an ETSP solution. Figures 10b and 10c show the results of increasing the radius to 20 for two different search algorithms. As the radius increases, the solution path do not take the form of ETSP solutions.

For very large radii, the TSPJointRelaxedD/AA algorithm outperforms (Figure 9). In fact, the algorithms that have the Alternating Algorithm within the search, such as the ETSP/AA Search and the TSP AA Search, benefit greatly for very large radii. These algorithms produce paths that are on average twice as short, or twice as better, as other algorithms, as some of the other algorithms achieve a quality score of 50 compared to the alternating algorithms achieving a score of 100. The reason for this can be explained by examining the solution paths found by the algorithms, such as in Figure 10d. This trajectory came from the ETSP/RelaxedD algorithm. As the radius becomes very large, the solution path can be described as performing circles. With a very large radius, it becomes difficult for the vehicle to maneuver to nearby

targets, located in the 20 x 20 target area. Due to the lack of maneuverability, the vehicle is forced to move outside the target area and perform a nearly complete circle in order to orient itself such that it will fly over the next target. Thus, for every target, the vehicle is forced to perform a nearly complete circle. Hence in the figure, there are 6 circles, one for each target. In contrast, with the Alternating Algorithm, every other segment along the path, the vehicle is forced to orient itself such that it is pointing directly at its next target, requiring no maneuvering from the vehicle. See Figures 10e and 10f. Notice that there are only 3 circles for the 6 targets. This saves the distance of another vehicle circle. This is a "2 for 1" phenomenon, as each vehicle "pass" over the target area intercepts two targets. This explains why the algorithms consisting of the Alternating Algorithm concept can have solutions that are roughly twice as short as other algorithms.

Now, examine the data points for medium sized radii in Figure 9. It appears that the GTSP uniform (4) and GTSP straight line angles (4) slightly outperform the others. In order to examine this, see Figure 11, which shows the average cost (path length) vs the number of expanded paths for radius of 1.5 over the 100 trials. Recall that the number of expanded paths is the number of partial paths that were taken off the queue and not pruned. The algorithms complete when the number of expanded paths is at a maximum. A better algorithm will provide a lower average cost. The data is from the same data as Figure 9, but specifically examines when the radius is 1.5. From the figure, it is possible to see that, when the algorithms complete, the GTSP straight line angles (4) algorithm provides a better cost than the other algorithms because the average cost is lower than the other algorithms. However, the number of expanded paths required is much higher. In contrast, the TSPJointRelaxedDAASearch algorithm finds a shorter path given a smaller number of expanded paths. Thus, for this case, there is a trade off between the number of expanded paths, a computational cost, and the shortest path found by the algorithms.

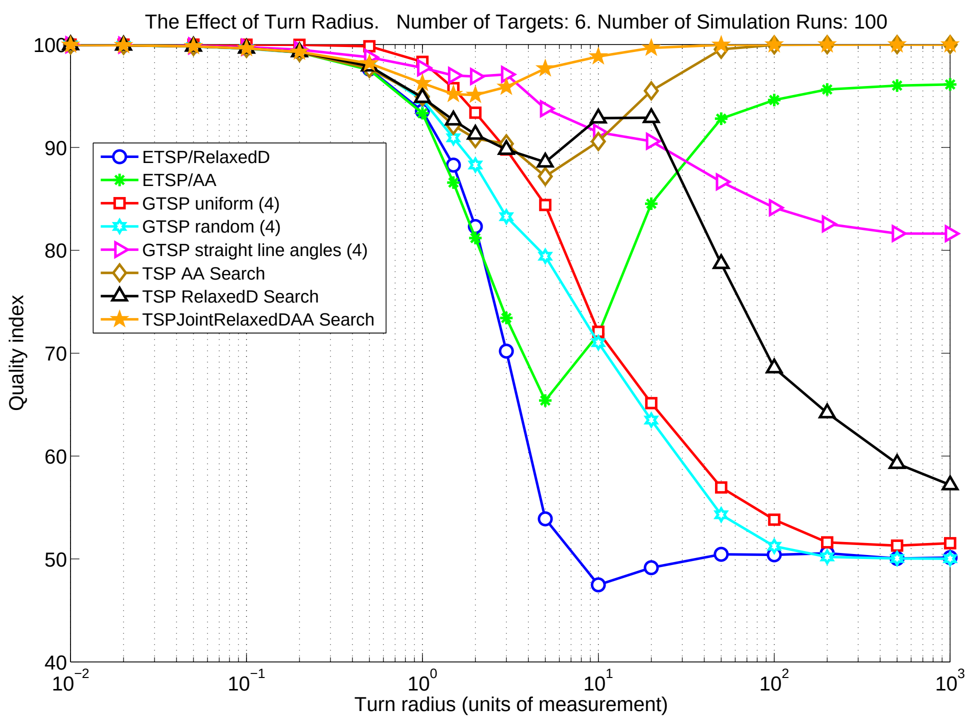
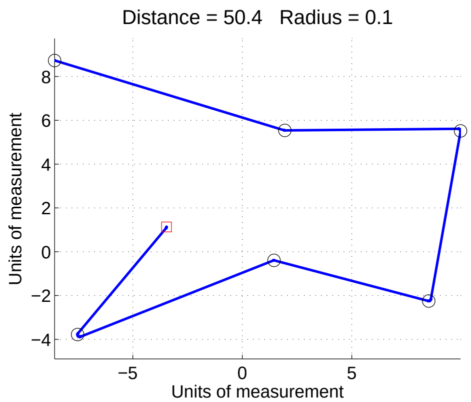


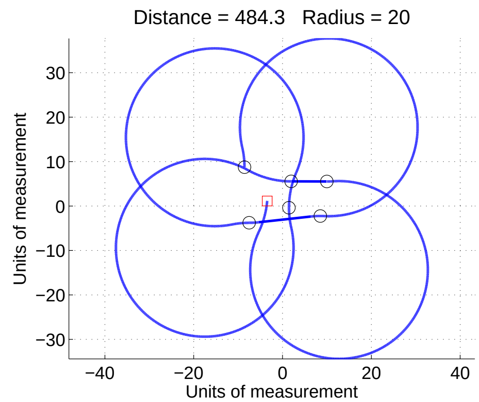
Figure 9. This figure shows a comparison between the 8 search algorithms as a function of turn radius for 6 targets located randomly in a 20 x 20 square. All search algorithms ran to completion.

IV.C.2. Analysis of algorithms that do not run to completion

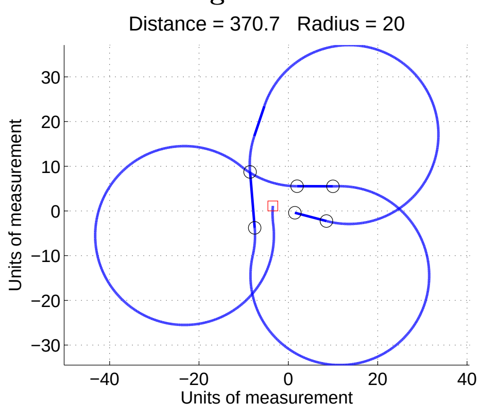
The following describes the results of stopping the algorithms after a runtime of 10 seconds for a dataset of 100 trials. Instead of 6 targets, 32 targets were randomly distributed in a 20 x 20 square. The results in Figure 12 show that the TSPJointRelaxedD/AA Search outperforms the other algorithms, especially as the



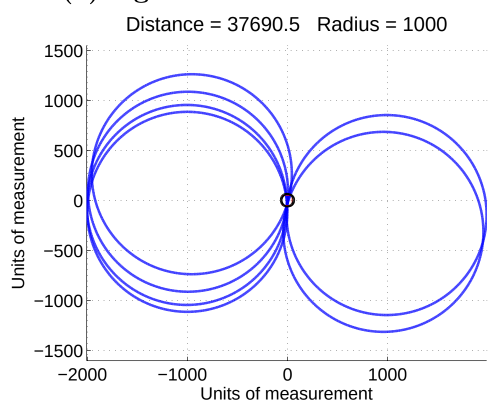
(a) All search algorithms provide solutions that converge to the ETSP solution



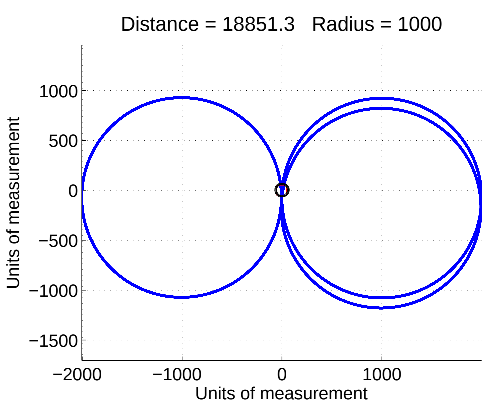
(b) Path generated by the GTSP uniform(4) algorithm



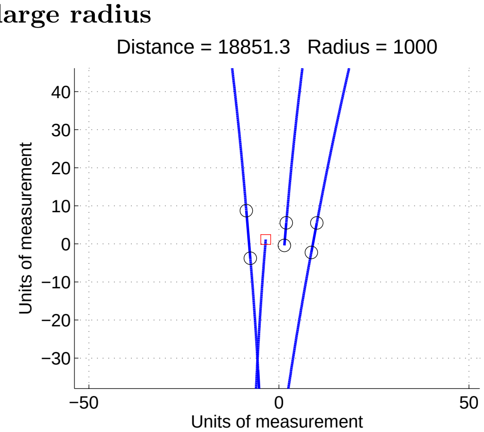
(c) Path generated by the TSP AA Search algorithm



(d) Path generated by the ETSP/RelaxedD algorithm for a very large radius



(e) Path generated by the TSP AA search algorithm for a very large radius



(f) Zoom-in of Figure 10e

Figure 10. Comparison of paths generated by different algorithms with different radii.

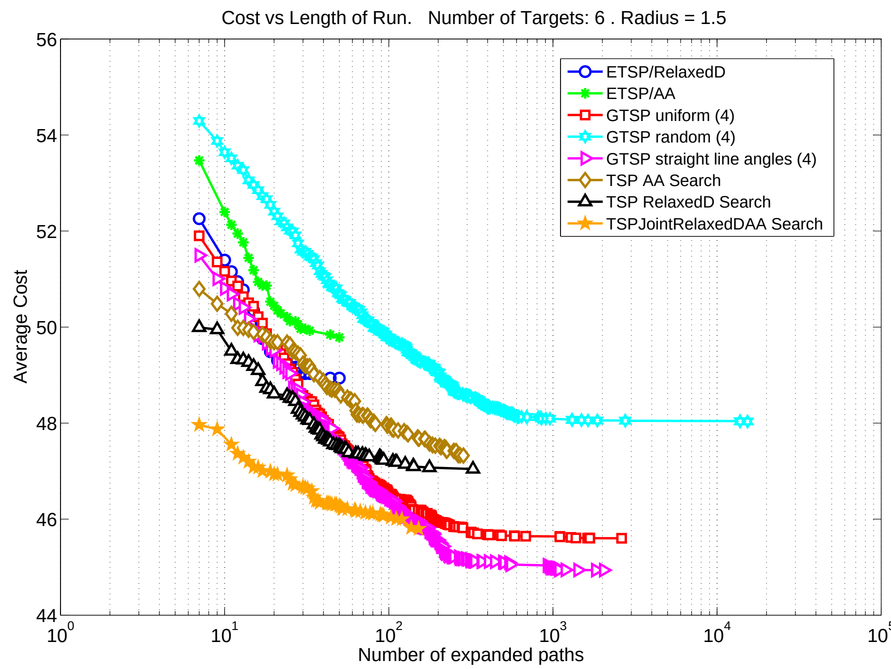


Figure 11. This figure shows the details of the simulation runs for a radius of 1.5. Though the GTSP straight line(4) eventually shows a lower average cost at the end, the cost of complexity is much higher. The TSPJointRelaxedDAASearch outperforms all other algorithms for the number of expanded paths checked.

radius of the vehicle increases. For small turn radii, the TSPJointRelaxedD/AA Search performs roughly the same as the other TSP algorithms. Again, this is due to the fact that the DTSP converges to the ETSP. Notice that the GTSP algorithms underperform for small turn radii. Because the DTSP converges to the ETSP for small radii, the orientation of the vehicle over each target is much less important. While the GTSP algorithms spend time searching through orientation possibilities as well as the ordering of the targets, the TSP algorithms search only for the best ordering of targets. The latter proves more successful for small turn radii.

When the radius is very large, the results are very similar to the results found for the situation in which the algorithms run to completion. For medium sized radii, the TSPJointRelaxedDAASearch outperforms, since the algorithms all run roughly the same number of expanded paths.

IV.C.3. Analysis of increasing the number of angles for GTSP algorithms

Figure 13 compares the GTSP uniform algorithm given different number of angle candidates. GTSP uniform(1), GTSP uniform(2), GTSP uniform(4), GTSP uniform(8) algorithms, assign potential angle headings over each vertex as $\{0\}$, $\{0, \pi\}$, $\left\{0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}\right\}$, and $\left\{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi, \frac{5\pi}{4}, \frac{3\pi}{2}, \frac{7\pi}{4}\right\}$, respectively. In this situation, clearly the larger number of angles dominates a lower number because all the angles of the smaller number are included in the higher number algorithm, and the algorithms are run to completion. This is shown in the figure as the number of winners for GTSP uniform(8) is at a maximum of 100. Similar to previous results, the percent difference between the algorithms, shown by the quality index, is small because the DTSP converges to the ETSP for small radii. When the radius is large, the paths converge to repeated circles. The major difference between the algorithms is in the middle range of the radii, where the solution paths to not converge to either the ETSP or repeated circle solutions.

Figure 14 shows the average cost of the solution vs turn radii for the different algorithms. As expected, larger radii will generally increase the length of the paths. For small radii, the difference between the average cost for the algorithms is small because of the convergence to the ETSP. For very large radii, the solutions all converge to repeated circles, leading to smaller differences between algorithms. In the middle range of

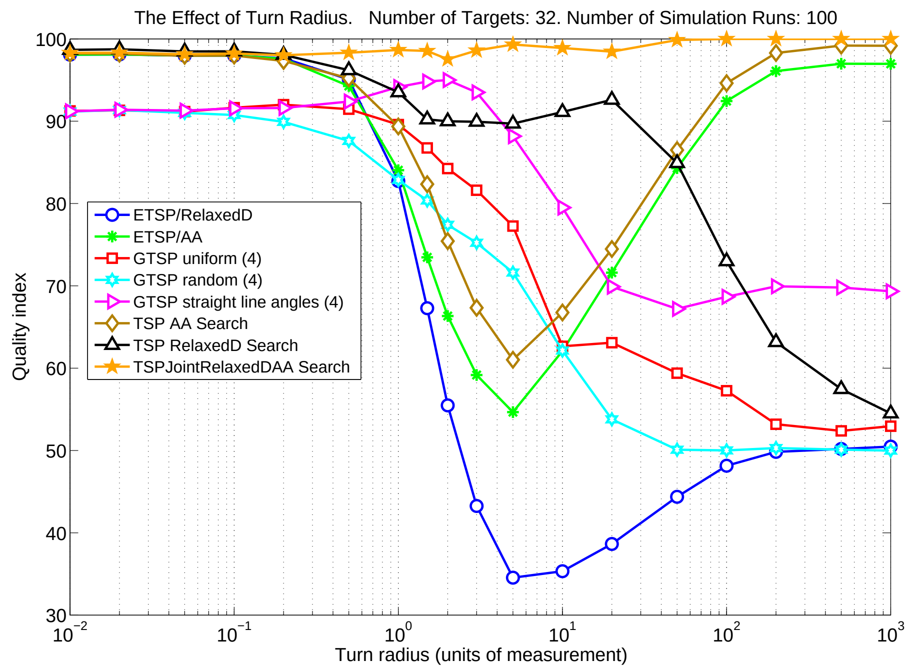


Figure 12. This figure shows a comparison between multiple search algorithms as a function of turn radius for 32 targets located randomly in a 20 x 20 square. All the algorithms were stopped before completion at 10 seconds. The TSPJointRelaxedDAASearch did the best or among the best for all radii.

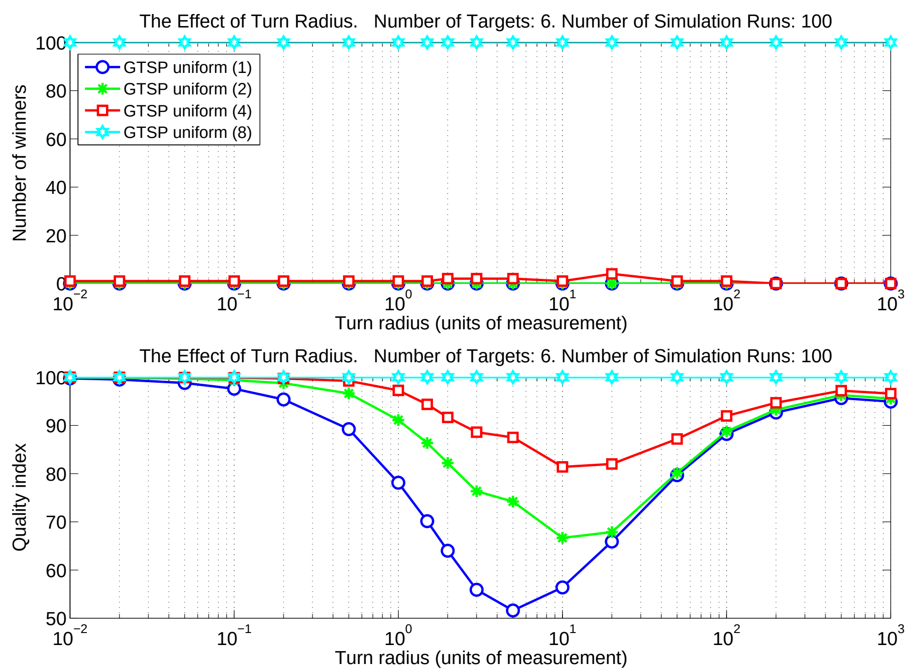


Figure 13. Comparison of GTSP uniform algorithm with different numbers of angles. As the number of angles increases, the solution improves, but at the cost of complexity. As expected, GTSP uniform(8) dominates the other algorithms.

the radii, the vehicle dynamics play a large role, and thus the difference in average costs for the algorithms is greatest here.

Figure 15 and Figure 16 show the results of the GTSP random algorithms, which are similar to the results of the GTSP uniform algorithms. Because the angles are generated randomly, the GTSP random algorithms with larger number of angle candidates do not necessarily dominate the same algorithm with fewer angle candidates.

For the dataset of 32 targets, the results of the GTSP uniform and GTSP random algorithms differ for smaller radii compared to the dataset of 6 targets. See Figures 17 and 18. For very small radii, fewer angle possibilities offer better solution. With fewer angle possibilities to check, the algorithm can search the ordering of vertexes more thoroughly. The order of vertexes is more influential in the cost than the angle at vertex as the turn radius approaches zero.

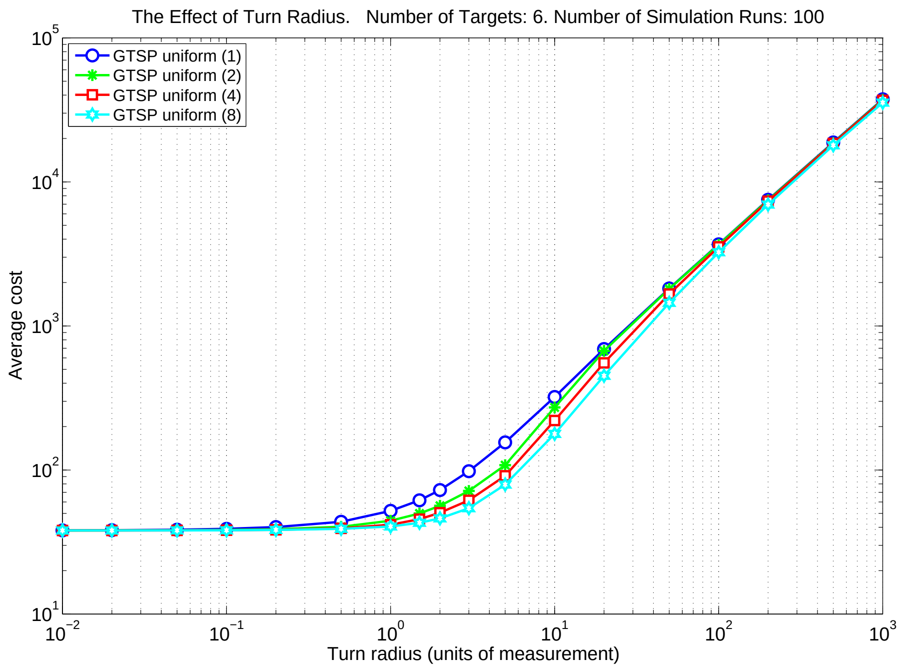


Figure 14. Comparison of GTSP uniform algorithm with different numbers of angles. The figure shows the average cost of the solution found as a function of the turn radii. As the radius increases, the average cost of the paths increase.

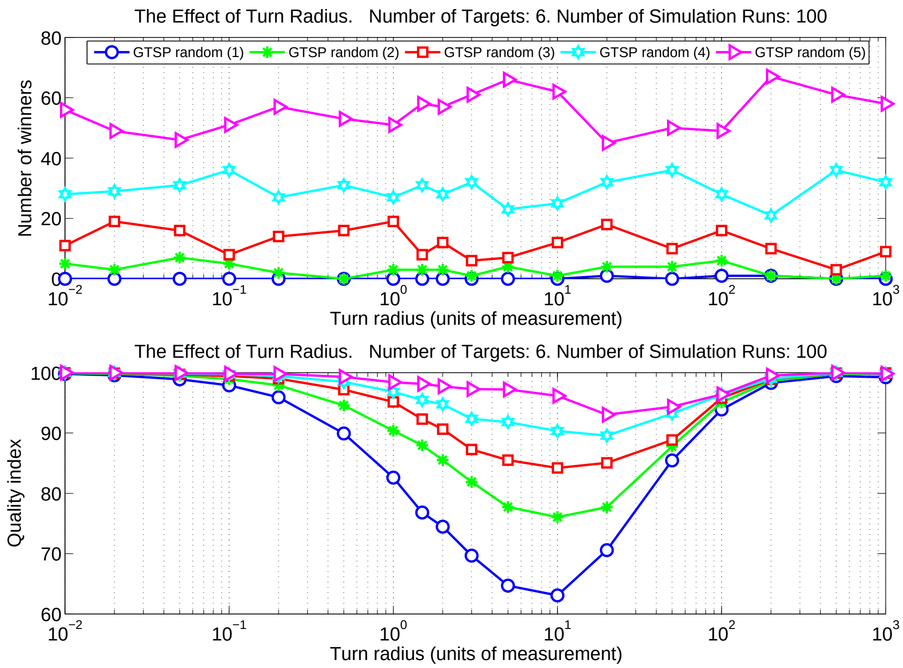


Figure 15. Comparison of GTSP random algorithm with different numbers of angles. Because the algorithms run to completion, the GTSP random(5) outperforms the same algorithm with fewer angle candidates.

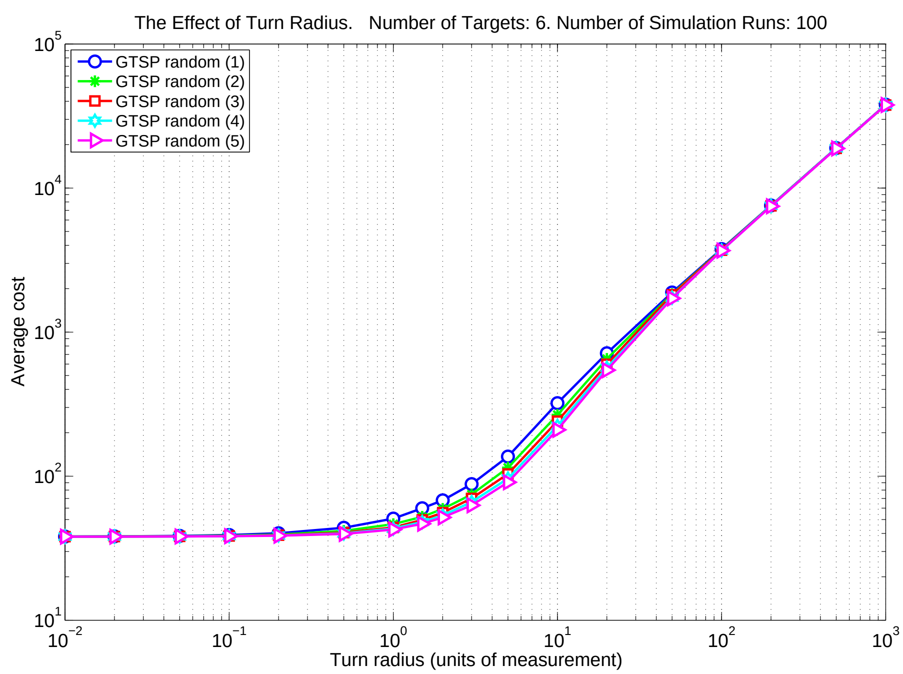


Figure 16. Comparison of GTSP random algorithm with different numbers of angles. The figure shows the average cost of the solution found as a function of turn radii. As the radius increases, the average cost of the paths increase.

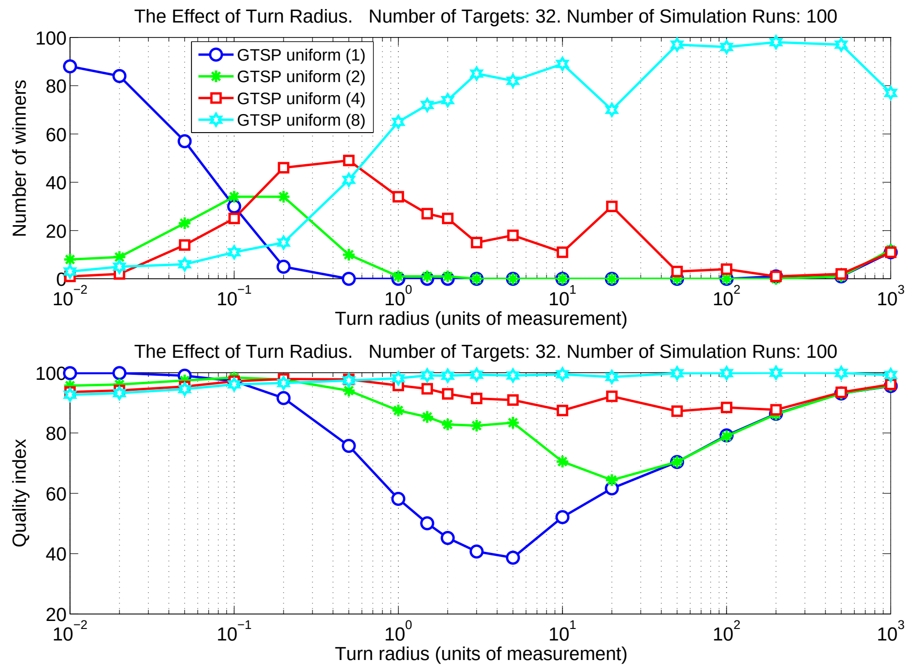


Figure 17. Comparison of GTSP uniform algorithm with different numbers of angles. When the algorithms do not run to completion, the GTSP algorithm with fewer angle choices outperform when the radius is small.

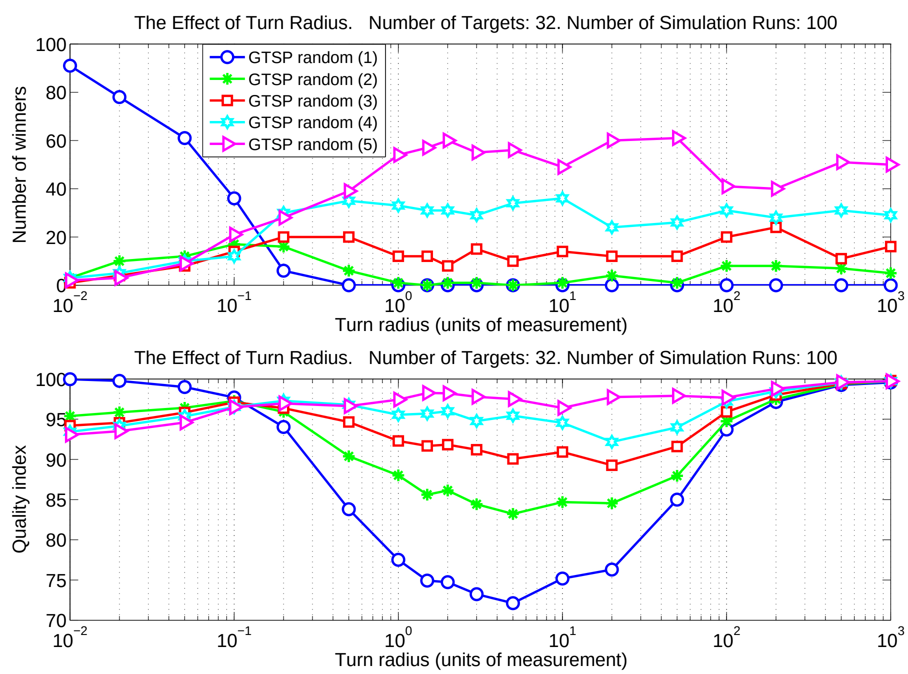


Figure 18. Comparison of GTSP random algorithm with different numbers of angles. When the algorithms do not run to completion, the GTSP algorithm with fewer angle choices outperform when the radius is small.

IV.D. Summary of simulation results

Among the methods tested, a TSP with a Dubins metric of cost-so-far, appears to be the best overall approach in terms of both complexity and best solution. As expected, as the radius approaches zero, the solutions reduce to the ETSP solution. As the radius approaches a very large value, the type of solution converges to performing circles to reach the targets. Because of this, the difference in the cost of the solution for very large radii becomes less significant over different algorithms. Furthermore, for very large turn radii, algorithms that incorporate the Alternating Algorithm do significantly better than the other algorithms.

V. Guidance through an Ordered Set of Targets

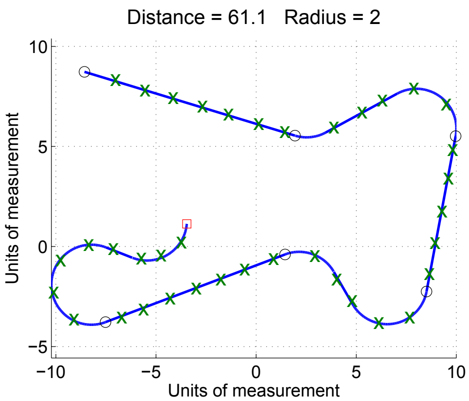
A UAV may need to traverse a particular trajectory or a particular ordering of targets obtained by a search algorithm illustrated in the previous section. This can be done by using an appropriate guidance law as outlined in Figure 1. Following a given trajectory can be accomplished by setting waypoints along the targeted trajectory, called waypoint guidance [17]. The vehicle's autopilot system is then responsible for flying to each waypoint. Possible methods include uniformly or non-uniformly placing waypoints along the trajectory (Figures 19a and 19b). It may be advantageous to add waypoints along highly curved segments in order that the vehicle will follow the path more closely during more difficult vehicle maneuvers. However, it may be possible to just use the targets themselves as waypoints instead of adding additional waypoints (Figure 19c).

The goal of this section is to address how a fixed-wing vehicle can be guided to follow a point to point Dubins trajectory, e.g., any trajectory found by a search algorithm in the previous section, with a minimum number of waypoints between targets using proportional navigation (PN) guidance law. PN [18, 19] is investigated because it is a well-known guidance law commonly used for guiding a vehicle to non-maneuvering targets . For a point to point Dubins trajectory, it will be shown that PN will converge to a bang-bang controller when sufficient number of waypoints are used or if the navigation gain of PN is very large. Because the task planner and motion planner can be implemented independently of the guidance law, the common PN guidance law is chosen for investigation over a bang-bang control law. For example, if a task planner and motion planner are chosen differently than as described in the previous sections, then a point to point Dubins trajectory may not be guaranteed, negating the need or desire for a bang-bang guidance law.

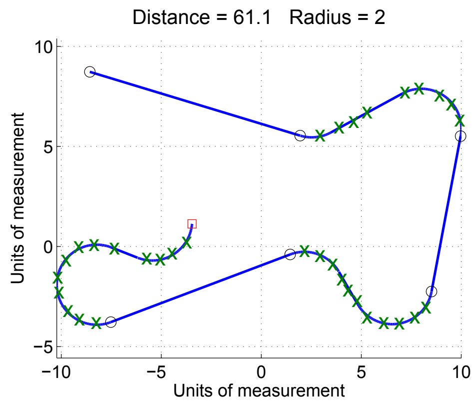
First, this section will review well-known results of PN. Then, the applicability of using PN for guiding a fixed-wing vehicle to a stationary target will be discussed. After limitations of using PN are shown, the use of appropriate waypoints is proposed and investigated to overcome these limitations and to ensure reachability to the target via PN. Finally, the concepts will be extended to include guidance to a point to point relaxed Dubins trajectory and a point to point Dubins trajectory, trajectories that result from the eight proposed algorithms discussed in the previous sections.

V.A. Traditional results of PN to intercept a stationary target

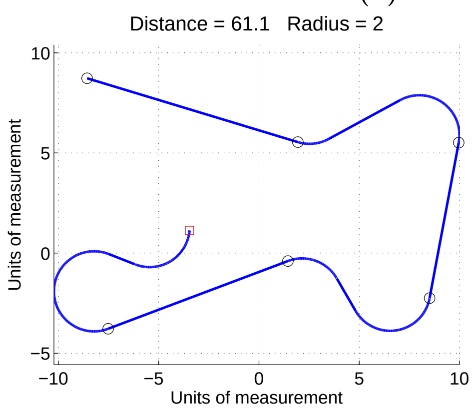
The goal of PN is to bring the pursuer to a collision course with the target. This is achieved when the change in the line of sight (LOS) angle between the pursuer and the target is zero. The engagement scenario between a vehicle and target is shown in Figure 20.



(a) Uniform distribution



(b) Nonuniform distribution



(c) The only waypoints are the targets

Figure 19. Waypoint guidance

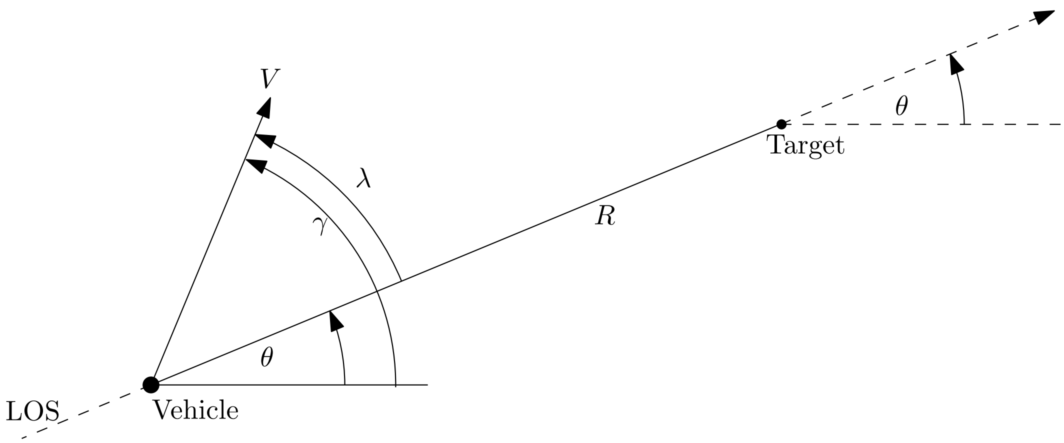


Figure 20. Engagement scenario

From Figure 20, the kinematics of the engagement are as follows. Let $\lambda \equiv \gamma - \theta$. V is the vehicle velocity, R is the range, θ is the line of sight angle. γ is the angle between the LOS and the velocity vector.

$$\dot{R} = -V\cos(\gamma - \theta) = -V\cos(\lambda) \tag{1}$$

$$\dot{\theta} = \frac{-V\sin(\gamma - \theta)}{R} = \frac{-V\sin(\lambda)}{R} \tag{2}$$

The proportional navigation law issues the following acceleration command

$$u = N'V\dot{\theta} = -\frac{N'V^2\sin(\lambda)}{R} \tag{3}$$

where N' is the navigation constant.

From the kinematic equations of the engagement and the acceleration command from the proportional navigation law, the following equations are derived by Shneydor [18]:

$$\frac{R}{R_0} = \left(\frac{\sin((N'-1)\theta + \lambda_0)}{\sin\lambda_0}\right)^{\frac{1}{N'-1}} \tag{4}$$

for $N' > 1$.

$$u = -\frac{N'V^2\sin\lambda_0}{R_0}\left(\frac{R}{R_0}\right)^{N'-2} \tag{5}$$

As Shneydor noted, the acceleration command is only bounded when $N' \geq 2$. When V is constant and $N' = 2$, the acceleration command is constant, and therefore the vehicle performs a circular trajectory. The acceleration command reduces to

$$u = -\frac{N'V^2\sin\lambda_0}{R_0} \tag{6}$$

V.B. Results using PN to intercept a stationary target

The results of implementing traditional PN on a stationary target are shown in Figure 21. The vehicle's maximum acceleration, velocity, and minimum turn radius R_{min} are equal to 1.

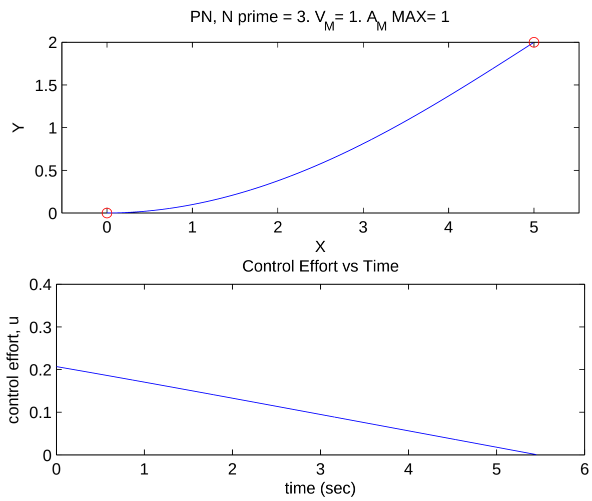


Figure 21. Standard PN to Target. $N' = 3$**. Maximum acceleration** $=$ **1. In the top graph, the vehicle travels from the origin** $(0, 0)$ **to the target at** $(5, 2)$**. The bottom graph shows the acceleration command vs time.**

However, when the target is located within the turn circle of the vehicle, then the position is "hard" because the vehicle cannot reach the target using PN, as shown in Figure 22. Notice that the control is saturated. The idea that a vehicle would not be able to reach a target within the vehicle's turn radius was noted by Robb et al., but without explicit proof or application to specific guidance laws [20].

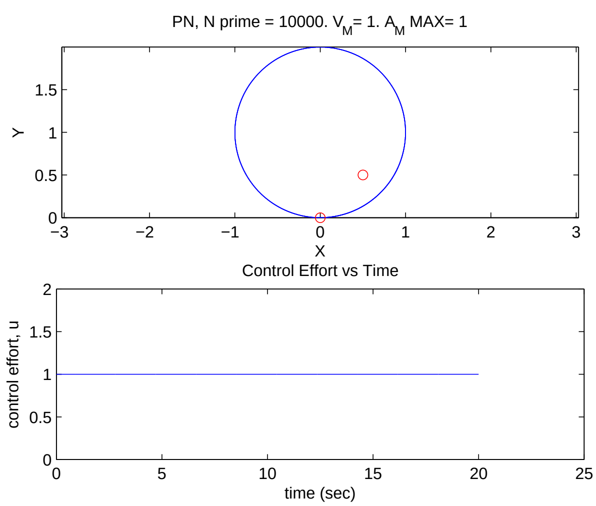


Figure 22. A vehicle starting from the origin with zero heading angle (pointing in the positive x axis) uses PN to travel to a target placed in a hard position. The vehicle does not intercept the target. Instead, it circles the target indefinitely, and the control is saturated.

Figure 23 illustrates the "hard" cases and the non-hard cases.

Lemma 1. *A target point* P (r_p, θ_p) *positioned in the interior of a vehicle's turn radius is not reachable by a vehicle implementing simple proportional navigation with an unchanging navigation gain,* $N' = 2$*. In other words, If* $R_{min} > |\frac{r_p}{2\sin\theta_p}|$ *and* $\gamma = 0$*, where* R_{min} *is the given minimum turn radius of the circle, and* θ *is the LOS angle, then the target is unreachable using simple PN.*

Proof. The minimum turn radius of a constant speed vehicle with a maximum acceleration can be derived by

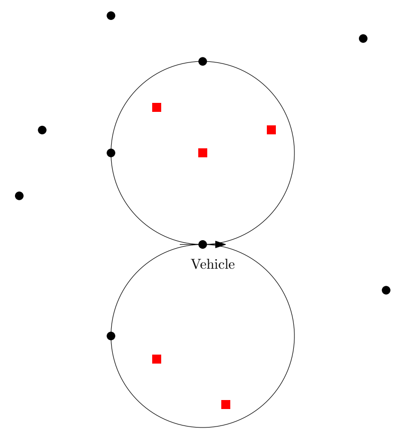


Figure 23. Potential targets using proportional navigation. The small squares located within the vehicle's turn circle are potential targets that are unreachable using simple PN. The small circles located on or outside the turn circle are potential targets that are reachable.

$$u_{max} = \pm \frac{V^2}{R_{min}} \tag{7}$$

where V, u_{max}, R_{min}, are the vehicle's constant speed, maximum acceleration, and minimum turn radius. The vehicle's maximum turn rate, $\dot{\gamma}_{max}$, is derived as follows

$$\dot{\gamma}_{\max} = \frac{u_{\max}}{V}$$

Without loss of generality, assume that the vehicle is located at the origin with a zero heading angle and would like to traverse to a target point P (r_p,θ_p), in which $0 < \theta_p < \pi$. See Figure 24. From Section V.A, when $N' = 2$ and the heading angle $\gamma = 0$, the acceleration is found from Equation 6, reproduced below:

$$u = -\frac{2V^2 \sin \lambda_0}{R_0}$$

Because $\gamma_0 = 0$, $\lambda_0 = -\theta_0$. Therefore,

$$u_{required} = \frac{-2V^2 \sin(-\theta_p)}{r_p} = \frac{2V^2 \sin \theta_p}{r_p} \tag{8}$$

Comparing 7 and 8, it is clear that

$$|u_{required}| > |u_{max}|$$

if

$$R_{min} > |\frac{r_p}{2 \sin \theta_p}|$$

When $u_{required} > u_{max}$, then the acceleration of the vehicle is saturated $u = u_{max}$. As a result, the vehicle will make circles of radius R_{min} with equation

$$R(\theta) \quad = \quad 2R_{min} \sin \theta \tag{9}$$

as long as the acceleration remains saturated. This is in fact the case because this circle encircles the target. As a result, at any point in the vehicle's trajectory, the vehicle and target can be viewed as being in the state described in Figure 24: the target is located within the turn radius of the circle such that the vehicle is located at the origin with a zero heading angle.

□

Lemma 2. *A target positioned in the interior of the vehicle's turn radius is not reachable by the vehicle by simple proportional navigation with an unchanging navigation constant, $N' > 2$.*

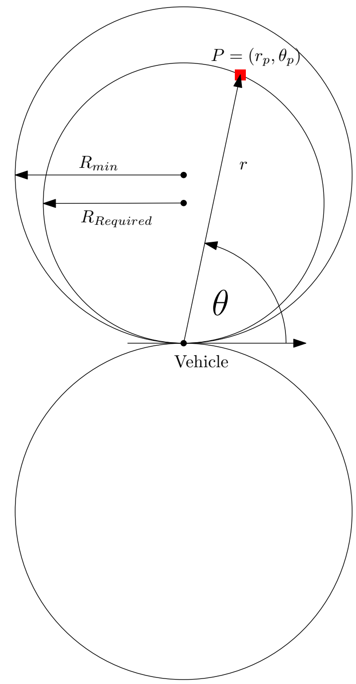


Figure 24. This figure shows the configuration of a vehicle and a target in polar coordinates, where the vehicle is located at the origin with $\gamma = 0$. The two large circles show the vehicle's minimum turn radius R_{min}. The target is located at point P (r_p, θ_p). The inner circle that intersects the target is the PN trajectory assuming no acceleration saturation and $N' = 2$. The current range is r_p. This figure graphs Equations 9.

Proof. From Lemma 1, we see that the saturation of the vehicle is reached when $N' = 2$ for the "hard" target problem. From Equation 3, a larger N' will demand larger acceleration. Since the acceleration is already saturated. The saturation command will be the same, i.e.

$$u_{N'>2} = u_{N'=2} = u_{max}$$

Thus the vehicle trajectory will be the same, that is, encircling the target indefinitely along the vehicle's turn circle. □

Lemma 3. *A target positioned on the vehicle's turn circle is reachable using simple proportional navigation for $N' \geq 2$, and the trajectory will be an arc of a circle along the vehicle's turn circle.*

Proof. This proof is similar to the proofs of Lemma 1 and Lemma 2. Without loss of generality and assuming the vehicle's state is that found in Figure 25, then the acceleration required, $u_{required}$, is as stated in Equation 8 for $N' = 2$. As can be seen from the discussion in Lemma 1, $u_{max} = u_{required}$ if $R_{min} = \frac{r_p}{2\sin\theta_p}$. One can verify this by examining the geometry of the problem (Figure 25). With a saturated acceleration, the vehicle will traverse along its turn circle as shown in Equation 9 until it reaches the target. Again, at any point in the vehicle's trajectory, the vehicle and target can be viewed as being in the state described in Figure 25: the target is located on the vehicle's turn circle such that the vehicle is located at the origin with a zero heading angle. Similar to Lemma 2, if $N' > 2$, then the acceleration demand will be higher, which will maintain a saturated acceleration, forcing an identical trajectory.

□

V.C. Relaxed Dubins and using waypoints to deal with "hard" problems.

It has been shown that targets located within the vehicle's turn circle are unreachable using simple PN with $N' \geq 2$. However, it was hypothesized that if a waypoint is created far away from the initial vehicle position and the target destination, then it will be possible to reach the target. However, in order to create a waypoint such that the trajectory will be in minimum distance, the relaxed Dubins problem is investigated.

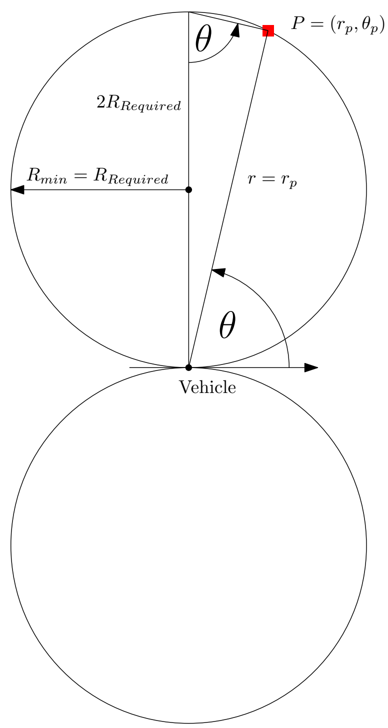


Figure 25. This figure shows the configuration of a vehicle and a target in polar coordinates, where the vehicle is located at the origin with $\gamma = 0$. The target is represented as a square located at (r_p, θ_p), which is on the vehicle's turn circle.

The relaxed Dubins problem is the problem of traversing from an initial position and orientation (x_i, y_i, θ_i) to a final position (x_f, y_f) without considering the final heading in minimum distance (and minimum time if the velocity is constant). This is similar to proportional navigation in that the vehicle starts with a specific (x_i, y_i, θ_i) and desires to arrive to a final (x_f, y_f) without considering the final heading; however, the relaxed Dubins problem is to achieve this in the minimum distance or time, for a constant speed vehicle, while the objective for PN is to reach the point while minimizing a running cost on the states and control. From the relaxed Dubins solution, the waypoint chosen is the point of transition between commands of turn left or turn right. See the bottom two trajectories of Figure 26. The waypoints are reachable because they are located on the vehicle's turn circle.

Figure 27 shows the result of using the relaxed Dubins analytical solution to generate the waypoint to reach a hard target. From the vehicle's initial position at the origin, PN is used to navigate to the waypoint. PN is used again to reach the final destination at $(0.5, 0.5)$. Notice that the acceleration is always saturated. The trajectory found is the relaxed Dubins solution.

V.D. Proportional navigation guidance as relaxed Dubins paths

Lemma 4. *For a vehicle with a minimum turn radius using proportional navigation guidance to reach a non "hard" stationary target, a relaxed Dubins path will be achieved as the navigation constant goes to infinity.*

Proof. For non "hard" problems: From Equation 5, it is clear that as N' gets very large, the acceleration command, u, will increase in magnitude. For a vehicle with a maximum acceleration limit u_{max}, $u = u_{max}$ as N' approaches infinity, unless $\gamma = \theta$. Thus, the vehicle will maintain a constant maximum acceleration, producing a circular trajectory, until $\gamma = \theta$. When $\gamma = \theta$, the vehicle is heading directly to the stationary target and no further acceleration command is required. Thus, the path consists of a circular arc followed by a straight line. The results of a very large N' and $N' = 3$ are shown in Figures 28a and 28b, respectively. Notice that bang-bang control is implied for very large N'.

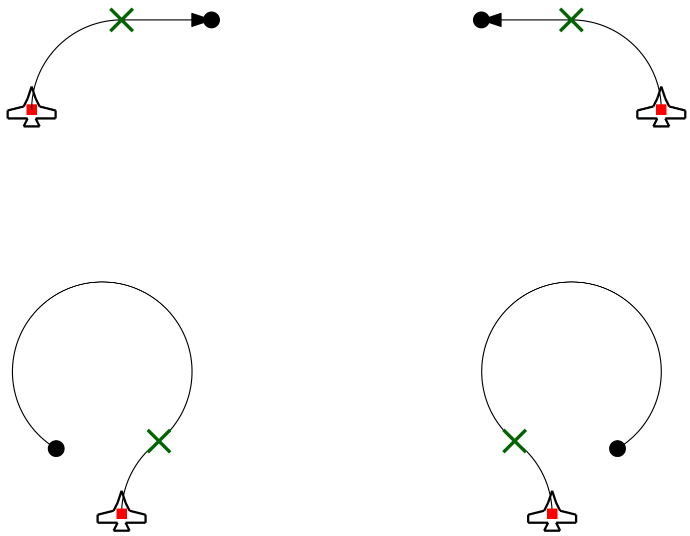


Figure 26. The four relaxed Dubins solutions with proposed waypoints shown as an 'x' at the transition between acceleration commands.

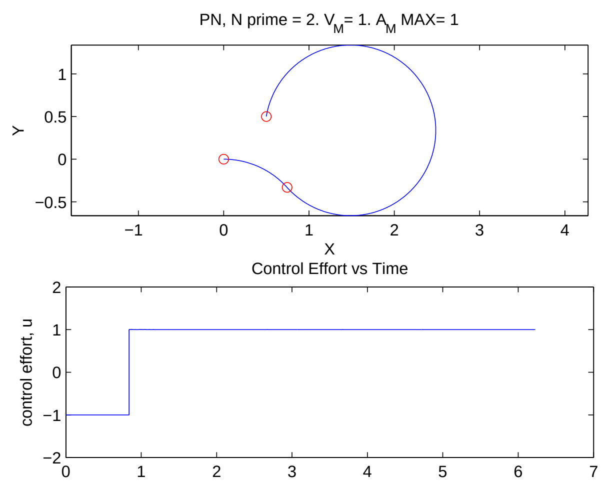
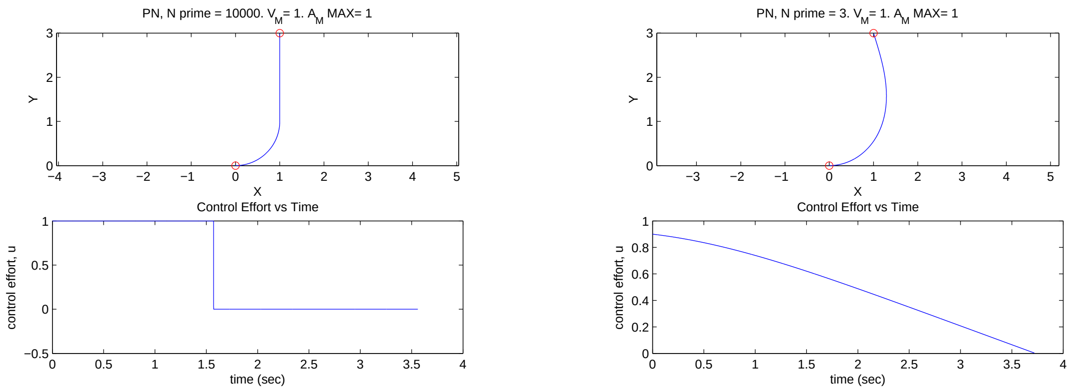


Figure 27. Result using a waypoint generated from the relaxed Dubins trajectory. The vehicle starts from the origin, goes through a waypoint, and finally reaches its target at point $(0.5, 0.5)$. The control is first saturated at $u = -1$ and then to $u = 1$, as the vehicle makes a right turn, then left turn maneuver.



(a) Very high N' (b) $N' = 3$

Figure 28. Results of using PN for a non "hard" target

$\square$

V.E. Results of proportional navigation and relaxed Dubins solution for traversing a set of ordered vertexes

The result of implementing proportional navigation with a very large navigation constant to multiple targets using the appropriate waypoints as discussed in the lemmas above is shown in Figure 29. The vehicle starts at the origin and traverses to the non hard target at $(-2, 1)$. Because of the vehicle's current orientation and position of the next target $(-2.5, 1.5)$, this next target is a “hard” target. After reaching this “hard” target, the vehicle finally arrives to the last target $(0, 1.5)$, a non hard target. The control takes values of fully saturated at ± 1 and zero control, corresponding to circle trajectories and straight line trajectories, respectively. Because the trajectory consists of point to point relaxed Dubins trajectories, it is similar to solutions generated by the search algorithms that use the relaxed Dubins algorithm as shown in the previous sections.

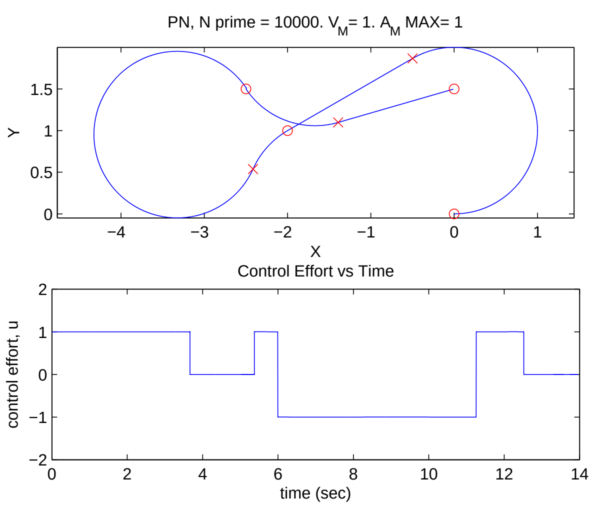


Figure 29. Result for traversing from the origin to 3 targets marked as circles. The navigation constant is very large ($N' = 10000$). The 'x' marks locations in which there is a fundamental change in control effort.

V.F. Proportional navigation and Dubins trajectory

It has been shown till now that it is possible to traverse any point to point relaxed Dubins trajectory, such as the ones derived in the search algorithms from the previous sections. This was done using waypoints along

the vehicle's turn circle from the relaxed Dubins solutions. The results can be extended easily to point to point Dubins trajectories, which were also derived in some of the search algorithms, by choosing waypoints along the transition points, shown as an 'x' in Figure 30.

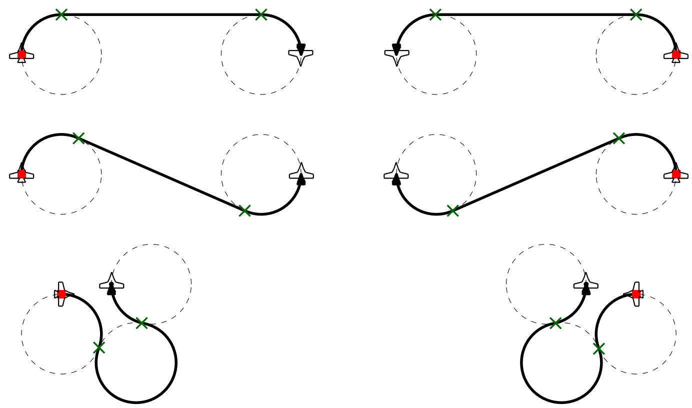


Figure 30. The six Dubins solutions with proposed waypoints shown as an 'x', located on the transition between acceleration commands.

V.G. Minimum waypoints required for relaxed Dubins solution trajectories

Recall that there are four possible relaxed Dubins solution trajectories including RS, RL, and their mirror images LS and LR. This section will discuss the minimum number of waypoints required in order for a Dubins vehicle to traverse a relaxed Dubins trajectory from an initial start position and under what conditions.

Lemma 5. *For the relaxed Dubins trajectory RS and LS, one waypoint, located at the end of the trajectory, is required when the vehicle is guided using simple proportional navigation with N' approaching infinity.*

Proof. This follows directly from Lemma 4. □

Lemma 6. *For the relaxed Dubins trajectory RS and LS, two waypoints, one located at the transition point and one located at the end of the trajectory, are required when the vehicle is guided using simple proportional navigation with $N' \geq 2$.*

Proof. The RS and LS relaxed Dubins trajectories are each composed of a circular arc along the vehicle's turn circle and a straight line. A waypoint located at the transition point will allow the vehicle to follow the first segment of the vehicle. Because the waypoint is located on the vehicle's turn circle, a value of $N' \geq 2$ will achieve this, as proven from Lemma 3. The second segment of the trajectory is a straight line. Since no acceleration command is required, $N' \geq 2$ will achieve the second segment. □

Lemma 7. *For the relaxed Dubins trajectory RL and LR, two waypoints, one located at the transition point and one located at the end of the trajectory, are required when the vehicle is guided using simple proportional navigation with $N' \geq 2$.*

Proof. The RL and LR relaxed Dubins trajectories are each composed of two circular arcs along the vehicle's turn circle. A waypoint located at the trajectory transition point will allow the vehicle to follow the first segment of the trajectory. Because the waypoint is located on the vehicle's turn circle, a value of $N' \geq 2$ is permissible. Similarly, a waypoint located at the end of the trajectory will force the vehicle to travel along its turn circle. □

V.H. Minimum waypoints required for Dubins solution trajectories

Recall that there are 6 possible relaxed Dubins solution trajectories including RSR, RSL, RLR, and their mirror images LSL, LSR, LRL, respectively. This section will discuss the minimum number of waypoints required in order for Dubins vehicle to traverse a Dubins solution trajectory from an initial start position and under what conditions. The logic follows that of the description of the minimum waypoints required for the relaxed Dubins solutions.

Lemma 8. *For the Dubins trajectories RSR, RSL, LSL, and LSR, two waypoints, one located at the second transition point and one located at the end of the trajectory, are required when the vehicle is guided using simple proportional navigation with N' approaching infinity.*

Proof. The RSR, RSL, LSL, and LSR Dubins trajectories are each composed of two circular arcs along the vehicle's turn circle and one straight line segment between the two. A waypoint located at the second trajectory transition point will force the vehicle to follow the first circular segment followed by the straight line segment. The second waypoint will guide the vehicle along the vehicle's turn circle to complete the path. □

Lemma 9. *For the Dubins trajectories RSR, RSL, LSL, and LSR, three waypoints, one located at the first and second transition points and one located at the end of the trajectory, are required when the vehicle is guided using simple proportional navigation with $N' \geq 2$.*

Proof. The RSR, RSL, LSL, and LSR Dubins trajectories are each composed of two circular arcs along the vehicle's turn circle and one straight line segment between the two. A waypoint located at the first transition will force the vehicle to fly along it's turn circle until it is in a straight line to the second transition point. A second waypoint located at the second transition will force the vehicle to fly along the straight line until a particular point. Then the vehicle will fly to the third waypoint along it's turn circle. □

Lemma 10. *For the Dubins trajectories RLR and LRL, three waypoints, one located at the first and second transition points and one located at the end of the trajectory, are required when the vehicle is guided using simple proportional navigation with $N' \geq 2$.*

Proof. The RSR, RSL, LSL, and LSR Dubins trajectories are each composed of three circular arcs. The proof follows that of Lemma 7.

Table 2. Number of waypoints required to perform a maneuver given a specific N'

	Relaxed Dubins		Dubins	
	RS, LS	RL, LR	RSR, RSL, LSL, LSR	RLR, LSR
$N' \geq 2$	2	2	3	3
$N' \to \infty$	1	2	2	3

□

Lemma 11. *For a trajectory only composed of point to point Dubins segments for a vehicle implementing PN with a finite $N' \geq 2$, the minimum number of waypoints, excluding the initial and final positions, required to follow the trajectory is equal to the number of transitions between turning right, turning left, and going straight.*

Proof. Vehicles implementing PN with a finite $N' \geq 2$ to a single stationary target or waypoint can only turn right, turn left (not necessarily along the vehicle's minimum turn radius), or go straight. It cannot, for example, turn right then turn left without adding additional waypoints. Thus the number of waypoints required to follow the trajectory is equal to the number of transitions between turning right, turning left, and going straight.

□

Lemma 12. *For a trajectory only composed of point to point Dubins segments for a vehicle implementing PN with N' approaching infinity, the minimum number of waypoints, excluding the initial and final positions, required to follow the trajectory is equal to the number of transitions of commands to turn right and turn left. Add one if the first maneuver is go straight.*

Proof. The set of maneuvers that a vehicle implementing PN with N' approaching infinity to a single stationary target or waypoint is {S, R, L, RS, LS}, in which the turns are along the minimum turn radius. Thus, any straight maneuver that is preceded by a turn maneuver can be combined, requiring just one waypoint instead of two. This was shown earlier in the proofs for point to point Dubins solution and point to point relaxed Dubins solution. No other reduction in waypoints can be achieved from other combinations of vehicle maneuvers. Therefore, the minimum number of waypoints required to follow the trajectory is equal

to the number of transitions to turn right and turn left. In the special case when the trajectory starts with a straight line, i.e., the vehicle's initial heading angle is already pointing directly to its first target, the first target must be included as a waypoint. Thus, one must be added to the total number of waypoints in this case. □

VI. Summary

This work investigated the Dubins traveling salesman problem because of its applications to UAVs. A Dubins vehicle was selected to model a fixed-wing UAV because of similar kinematic constraints. This work proposed and contrasted several different deterministic search methods for solving the Dubins traveling salesman problem, algorithms which include overlaying kinematic-satisfying solutions on the corresponding ETSP, approximating the DTSP into a GTSP reformulation, and integrating an upper bound Dubins cost in a TSP search. The GTSP and the TSP search with Dubins cost-so-far algorithms are integrated approaches which combine the task planning and motion planning together. The results show that the TSP search with Dubins cost-so-far algorithms generally provide solution paths that are shorter than the other algorithms. The TSP search with Dubins cost-so-far has the added advantage that the complexity remains that of a TSP rather than expanding to a GTSP.

Since it was hypothesized that the turn radius has a large effect on the solution paths, the algorithms were tested using Monte Carlo simulations. As expected, with a small turn radius, the algorithms tend toward the ETSP. With a very large radius, the solution paths tend to repeated circles because of the lack of vehicle maneuverability. While there may be benefits of using certain methods over others, the complexity of the problem and ease of implementation of the algorithms must also be considered.

After a trajectory through multiple target locations has been proposed, a guidance law must be implemented. This research investigated proportional navigation guidance, an optimal guidance law for non-maneuvering targets. It was shown that by using simple proportional navigation guidance, reaching a stationary target within the vehicle's turn circle is not possible for $N' \geq 2$. Furthermore, when the target is located outside the vehicle's turn circle, proportional navigation guidance was shown to tend toward the relaxed Dubins solution as the navigation constant approaches infinity. It was shown that when the target is within the vehicle's turn circle, a waypoint can be created in order to reach the target in minimum distance by using the relaxed Dubins solution. It was also shown that waypoints can be used to follow a point to point Dubins trajectory, such as the ones derived by the investigated search algorithms.

References

[1] Changchun, L., Li, S., Hai-bo, W., and Tianjie, L., "The research on unmanned aerial vehicle remote sensing and its applications," *Advanced Computer Control (ICACC), 2010 2nd International Conference on*, Vol. 2, March 2010, pp. 644 –647.

[2] Kontitsis, M., Valavanis, K. P., and Tsourveloudis, N., "A UAV vision system for airborne surveillance," *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, Vol. 1, April-1 May 2004, pp. 77 – 83 Vol.1.

[3] Erdos, D. and Watkins, S. E., "UAV Autopilot Integration and Testing," *Region 5 Conference, 2008 IEEE*, April 2008, pp. 1 –6.

[4] Murray, R. M., "Recent Research in Cooperative Control of Multivehicle Systems," *Journal of Dynamic Systems, Measurement, and Control*, Vol. 129, No. 5, 2007, pp. 571–583.

[5] Savla, K., Frazzoli, E., and Bullo, F., "Traveling Salesperson Problems for the Dubins Vehicle," *Automatic Control, IEEE Transactions on*, Vol. 53, No. 6, July 2008, pp. 1378 –1391.

[6] Dubins, L. E., "On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Position," *American Journal of Math*, Vol. 79, 1957, pp. 497–516.

[7] LaValle, S. M., *Planning Algorithms*, Cambridge University Press, Cambridge, U.K., 2006, Available at http://planning.cs.uiuc.edu/.

[8] Ma, X. and Castañón, D. A., "Receding Horizon Planning for Dubins Traveling Salesman Problems," *Decision and Control, 2006 45th IEEE Conference on*, December 2006, pp. 5453–5458.

[9] Enright, J. J., Savla, K., Frazzoli, E., and Bullo, F., "Stochastic and Dynamic Routing Problems for Multiple Uninhabited Aerial Vehicles," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 32, No. 4, July-August 2009, pp. 1152–1166.

[10] Savla, K., Frazzoli, E., and Bullo, F., "On the point-to-point and traveling salesperson problems for Dubins' vehicle," *American Control Conference*, Portland, OR, June 2005, pp. 786–791.

[11] Edison, E. and Shima, T., "Integrated task assignment and path optimization for cooperating uninhabited aerial vehicles using genetic algorithms," *Computers and Operations Research*, Vol. 38, No. 1, 2011, pp. 340–356.

[12] Gutin, G. and Punnen, A. P., *The Traveling Salesman Problem and Its Variations*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2002.

[13] Lawler, E. L., Lenstra, J. K., Kan, A. H. G. R., and Shmoys, D. B., *The Traveling Salesman Problem and A Guided Tour of Combinatorial Optimization*, Wiley, Chichester, 1985.

[14] Garey, M. R., Graham, R. L., and Johnson, D. S., "Some NP-complete geometric problems," *Proceedings of the eighth annual ACM symposium on Theory of computing*, STOC '76, ACM, New York, NY, USA, 1976, pp. 10–22.

[15] Papadimitriou, C. H., "The Euclidean travelling salesman problem is NP-complete," *Theoretical Computer Science*, Vol. 4, No. 3, 1977, pp. 237 – 244.

[16] Le Ny, J., Frazzoli, E., and Feron, E., "The curvature-constrained traveling salesman problem for high point densities," *Decision and Control, 2007 46th IEEE Conference on*, December 2007, pp. 5985 –5990.

[17] Shima, T. and Rasmussen, S. J., editors, *UAV Cooperative Decision and Control: Challenges and Practical Approaches*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1st ed., 2008.

[18] Shneydor, N. A., *Missile Guidance and Pursuit*, Horwood Publishing, Chichester, 1998.

[19] Zarchan, P., *Tactical and Strategic Missile Guidance*, Progress in Astronautics and Aeronautics, American Institute of Aeronautics and Astronautics, 2007.

[20] Robb, M., White, B. A., Tsourdos, A., and Rulloda, D., "Reachability guidance: a novel concept to improve mid-course guidance," *American Control Conference, 2005. Proceedings of the 2005*, June 2005, pp. 339 – 345 vol. 1.